MIKROSZÁMÍTÓGÉP MAGAZIN

A NEUMANN JÁNOS SZÁMÍTÓGÉP-TUDOMÁNYI TÁRSASÁG LAPJA

1986 szeptember

Ára: 30 Ft

Fiatalok! Figyelem!

Következő számunkban három hónapon át tartó rejtvénypályázatot indítunk! Ugyanakkor közöljük majd a megfejtések beküldési határidejét is. Ha több hibátlan megoldást kapnánk, kisorsoljuk azt a szerencsés nyertest, aki az alábbi — Spectrum számítógépekhez használható — kiegészítő eszközök közül kiválaszthatja a számára legmegfelelőbb nyereményt!

— INTELLIGENS JOYSTICK INTERFACE
— EPROMÉGETŐ és ELLENŐRZŐ (16K — 32K-hoz)
— KÜLSŐ PROGRAMTÁR
— UNIVERZÁLIS PRINTER INTERFACE
— FÉNYCERUZA
— MEMÓRIABŐVÍTÉS (16K → 48K)
— LAPOZÓS MEMÓRIA (48K → 80K)
— SPECTRUM — TON hangosító
— EGROMÉGETŐ (2716 → 27128)
— SPECTRUM ROM I. (módosított)

Fentiek megtekinthetőek a DIGITÁL Számítástechnikai Szaküzletben! Cím: 1026 Bp., Szilágyi Erzsébet fasor 35.

SOK SIKERT KÍVÁN A SZERKESZTŐSÉG!

MIKROSZÁMÍTÓGÉP MAGAZIN

4. ÉVFOLYAM
8. SZÁM
1986. SZEPTEMBER

A NEUMANN JÁNOS SZÁMÍTÓGÉP-TUDOMÁNYI TÁRSASÁG ÉS A KISZ KÖZPONTI BIZOTTSÁG LAPJA

A kiadvány a Tudományszervezési és Informatikai Intézettel együttműködve készül

A szerkesztőbizottság vezetője:
Kovács Győző

E számunkat szerkesztették:
Bakos Tamás (programozástechnika)
Broczkó Péter (hírek)
Kovács Győző (levelezés)
Lindner László (sakkprogramozás)
Petróczy Judit (könyvek)
Simonyi Endre (klub)
Varga András (iskola — számítógép)

PR menedzser:
Pálhalmi Vali

Felelős szerkesztő:
Könyves Tóth Pál
Szerkesztőség:
1027 Budapest II., Fő u. 68.
Telefon: 154-250

Kiadja az Ifjúsági Lap- és Könyvkiadó Vállalat
Felelős kiadó:
dr. Petrus György
igazgató
Kiadóhivatal:
1065 Budapest VI., Révay u. 16.
Telefon: 116-660

Terjeszti a Magyar Posta
Előfizethető a hírlapkézbesítő hivataloknál
és a Posta Hírlapelőfizetési és Lapellátási Irodáján
(1900 Budapest V., József nádor tér 1.)
vagy átutalással a 215—96 162 pénzforgalmi jelzőszámra.
Megjelenik havonta
Egy szám ára 30,— Ft
Előfizetési díj:
egy évre 360,— Ft
fél évre 180,— Ft
Külföldön terjeszti a Kultúra,
1389 Budapest, pf. 149.
és a Magyar Média
1932 Budapest, pf. 279.
86—253

Szikra Lapnyomda
Budapest (86-3721)
Felelős vezető:
Csöndes Zoltán vezérigazgató

INDEX: 25 629
ISSN 0236-6088

Címképünk:

A KODEX 2000 szövegszerkesztő

Tartalom

µ'87	2
A minőségügy — közügy	12
Assemblerek, cross-assemblerek	13
A CAD alapelemei Commodore mikroszámítógépeken	21
Helyi hálózatok	24
Adok—veszek—cserélek	31
Barátunk, a számítógép	32
Ki ad magyarázatot?	40
Külföldön olcsó — belföldön drága?	46
ISKOLA — SZÁMÍTÓGÉP	
A Commodore 16 mint iskolaszámítógép	3
Az ismeretlen C16	3
Az 1986. évi „Nemes Tihamér" középiskolás számítástechnikai versenyről	5
DIÁKROVAT	
A HT—1080Z lemezrendszer utasításai és függvényei	8
Amit nem tud a HT-val	9
Nagy felbontású grafika	10
PROGRAMOZÁSTECHNIKA	
BASIC és gépi kód	18
Z80 programozási gyakorlatok	19
µPROGRAMOK	
Sztringváltozóban elhelyezett összefüggést kiértékelő szubrutin	27
GET * helyett INPUT *	29
Beviteli rutin	30
UNIN példaprogram	31
µKLUB	
Az Építsünk számítógépet sorozat egy hibájáról	33
PTA—4000 BASIC	34
Forgástestek és felületek	35
TURBO interfész ZX—Spectrumhoz	38
SAKKPROGRAMOZÁS	
Bitek és figurák	39
JÁTÉKPROGRAMOK	40
AZ OLVASÓ ÍRJA	42
FÓRUM	44
KÖNYVEK	45
HÍREK, ÉRDEKESSÉGEK	46

Még egyszer a μ '87-ről

„Az alkotó gondolkozás általában kérdések, feladatok, problémák felmerülésekor indul meg. Ilyen értelemben nemcsak az egész emberiség számára új felfedezéseket, művészi teljesítményeket stb. kell alkotásnak minősítenünk, hanem a mindennapi élet mindenkor adódó, az egyén számára új, apró problémáit is, pl. egy bevásárlási program tervezését (milyen sorrendben és milyen járműveken látogatjuk sorra az üzleteket a legrövidebb úton), egy erősítő tervezését, ahol csak az erősítés és a rezgésszámhatár más, mint az eddig ismert kivitelekben, egy reprodukáló művész feladatát stb."

(Nemes Tihamér: Kibernetikai gépek)

A kedves olvasó joggal gyanakodhat, hogy eltévesztettem a dátumot és tíz hónappal korábban írtam meg a szerkesztőségi cikket. A kedves olvasó téved, az írás szándékos, a második Országos Mikroszámítógépes Találkozónak az ideinél jobb és hatékonyabb előkészítését szolgálja.

A μ'86-ról szóló hivatalos és nem hivatalos értékelések megtisztelő módon elismerőek voltak, ezért különös gondossággal szeretnénk a jövő évi találkozóra felkészülni, 1987-ben egy, az ideinél jobb és még több érdeklődőt vonzó programot szeretnénk összeállítani, ehhez kérjük az informatika alkalmazása iránt érdeklődők segítségét és támogatását.

Talán egyetértenek velem, hogy ismét, immáron harmadszor is meg kell rendeznünk „A számítástechnika mindenkiért, a számítástechnika mindenkié" kiállítást. A célt nem szeretnénk változtatni, ezen a kiállításon azokat az eszközöket, programokat, oktatási anyagokat, módszereket mutatjuk be, amelyeket kevés tanulással, kevés szakértelemmel bárki gyorsan alkalmazhat. Az idén a szakma elismert nagy intézményei mellett sok kisvállalkozás, szövetkezet, leányvállalat és gazdasági munkaközösség vonult fel, főleg a fővárosból. Jövőre azt szeretnénk, ha a vidék is méltóan képviselve lenne a kiállításon. Úgy gondoljuk, hogy az NJSZT megyei területi szervezetei mutassák be a helyi vállalkozások eredményeit és így a μ'87 egyben országos tapasztalatcserére is lehetőséget ad.

Változatlanul reméljük, hogy a társadalmi programok mellett meg tudjuk rendezni a szocialista országok első elektronikai vagy inkább informatikai szakkiállítását is. A tervezett együttműködés a Tavaszi Fesztiválért felelős IPV-vel, a kiállítások megszervezésében gyakorlott HUNGEXPO-val, az elektronizálási programért felelős főhatóságokkal és természetesen az érdekelt intézményekkel és vállalatokkal nem utolsósorban a Neumann János Számítógéptudományi Társaság szakembereivel egészen biztosan tovább növeli a kezdeményező Tavaszi Fesztivál tekintélyét.

Jövőre is meghívjuk az egészségkárosultak szervezeteit a kiállításra. Úgy gondoljuk, már az idén is bizonyítottuk, hogy a vakok és csökkentlátók, a mozgássérültek a számítástechnikában teljes emberként dolgozhatnak. Ma az a legnagyobb probléma, hogy erre viszonylag kevés a lehetőségük. Az igazi megoldás az volna, ha otthoni munkahelyeket lehetne kialakítani és a mozgásukban korlátozott emberek távbeszélő vonalra kapcsolt készülékekkel tudnák munkájuk eredményét a munkahelyükre küldeni. Egy távkapcsolatra alkalmas otthoni számítógéppel egyszerre több munkahellyel is összeköttetést lehet tartani, így az egészségkárosult emberek kevésbé éreznék, hogy kiszolgáltatott helyzetben vannak.

A vártnál sokkal több érdeklődőt vonzott az oktatási és általában az iskolai alkalmazási bemutató. Tanárok és diákok nagyszámú nézőközönségének mutatták be, hogy a számítógép nagyon rövid időn belül az iskolai oktatás nélkülözhetetlen segédeszközévé fog válni. Az általános iskolák gépesítése most kezdődik, a középiskolákban is egyre több tanár és diák vált az informatika „tudósává", ezért valószínűnek tartom, hogy a jövő évi bemutatók során is sok érdekes alkalmazást láthatunk.

Szeretnénk újabb pályázatokat kiírni és versenyeket indítani. Feltételezzük, hogy a hardver és szoftver amatőrök részben továbbfejlesztik rendszereiket, részben újakat terveznek, így a 87-es munkák legalább olyan sikert aratnak, mint az ideiek.

A napi programot ankétokkal, kerekasztal megbeszélésekkel zárjuk, felhasználva az alkalmat a számítástechnika aktuális problémáiról való véleménycserére.

Jövőre — talán ez lehetne a μ'87 jelmondata is — többet szeretnénk foglalkozni az informatika és a művészet kapcsolatával. Az a hangverseny, amelyet az idén az NJSZT zenei bizottsága rendezett, emlékezetes szép sikert aratott. A zenei bemutató nagyon sok érdeklődőt hozott ki a kiállításra, azok is eljöttek, akiket a számítástechnika kevésbé, inkább a mai modern zenei törekvések érdekeltek. A Budapesti Tavaszi Fesztivál rendezőinek azt javasoltuk, hogy 87-ben a fesztivált egy számítógépes-zenei hangversennyel lehetne nyitni, amelyen részben magyar szerzők, részben külföldiek — elsősorban Boulez — műveit mutatnák be. El szeretnénk hozni és meg is mutatni azokat a számítógépeket, amelyeket komponálásra, illeve zeneművek lejátszására konstruáltak.

A másik tervezett esemény a számítógéppel készült grafikai alkotások bemutatása. Az a szándékunk, hogy nemcsak hazai, de külföldi alkotók munkáit is kiállítjuk, ezzel is bizonyítva, hogy a számítógép nemcsak a tudomány és az üzleti élet, hanem a művészek számára is újabb lehetőségeket kínál.

Biztosan sokan ismerik a kibernetikus szobrokat, másnéven idő- és térstruktúrákat, amelyek, ha úgy tetszik, informatikai alkotások. Szeretnénk bemutatni hazai képzőművészeink alkotásai mellett például Nicolas Schöffer magyar származású francia művész munkáit, aki a világon talán legelőször készített ilyen műveket.

Nagyon örülnénk, ha sor kerülhetne más múzsáktól ihletett alkotók (pl. irodalmárok) bemutatkozására is, nagyon szívesen fogadnánk újabb javaslatokat.

Ennek az írásnak bevallott célja, hogy a μ'87 megrendezéséhez munkatársakat találjunk, újabb ötleteket kapjunk, hiszen csak így válhat az esemény azzá, aminek a neve után válnia kell, a mikroszámítógépet alkalmazók igazi országos fesztiváljává.

KOVÁCS GYŐZŐ

A Commodore 16 mint iskolaszámítógép

Lapunkban vita indult a C16 körül. Több olyan kérdés is felvetődött, amelyre nem közismert a válasz. Hogy a vitát szűkebb mederben tartsuk, most ilyen kérdésekre válaszolunk.

Hogyan lett a C16-ból iskolaszámítógép? A terveknek megfelelően a Tudományszervezési és Informatikai Intézet már 1984-ben elkezdett foglalkozni az általános iskolák számítógéppel való felszerelésével. A középiskolákban elterjedt HT-1080Z nem jött számításba, részben a szükséges mennyiség, részben ára, részben pedig szolgáltatásai miatt. A piacon akkor hozzáférhető áron, megfelelő szolgáltatásokkal csak a Primo állt rendelkezésre. Az iskolákban az első gyártási sorozat példányaival végzett kísérletek azonban megbízhatósági problémákat vetettek fel.

Ekkor kapóra jött egy ajánlat Commodore 16 gépek szállítására. A beszerzés nem igényelt devizát — a művelődési tárcának erre a célra nem volt devizája —, és a gépek minden addiginél olcsóbbak voltak, ami azt jelentette, hogy a rendelkezésre álló keretből jóval több ilyen számítógép vásárlására tellett, mint amennyire korábban gondolni lehetett. A szakértők szerint a gépnek nem volt olyan hátránya, amely indokolttá tette volna a rendkívül kedvező ár-teljesítmény viszony elutasítását. Így került be 2000 db C16 az országba. A devizakitermelést a Skála-Coop vállalta magára.

A gépeknek volt magyar ékezetes betűkészletük, de nem a nálunk szokásos elrendezésű billentyűzeten. A magyar nyelvű felhasználói kézikönyv az eredeti angol alapján készült, amely sajnos nagyon sok hibát tartalmazott. Ezeknek csak egy részét sikerült kiszűrni a rendelkezésre álló rövid idő alatt. A többször hivatkozott kapcsolási rajz, monitorleírás a német kiadásban található, amelyet akkor nem sikerült beszerezni. Sajnálatos módon az eredeti kézikönyvben — és így a magyar nyelvűben is — hivatkozott *Programozói referencia kézikönyv* sokáig nem jelent meg angolul sem. Jelenleg kézirata alapján fordítják, de sajnos a legigényesebb felhasználókat ez sem fogja teljesen kielégíteni. Ezért készül az ismert Becker-sorozatból a C16 ROM listáját is tartalmazó könyv fordítása is, mindez a Novotrade Rt-nél.

Az iskolák a gépeket a hozzájuk tartozó Datasette 1531 kazettás tárolóval, 3 db *Felhasználói kézikönyv*vel, 1 db bemutató kazettával (két játék-, egy rajzoló- és egy sakkprogram), ezek leírásával, egy a C16 BASIC nyelvébe bevezető önképző tanfolyami jegyzettel (*Bevezetés a BASIC nyelvbe*, 1. rész. Novotrade) és az ehhez tartozó 2 db kazettával, 1 db *Hetedhét Commodore 16* könyvvel (Révbíró—Pál, Novotrade), valamint a gépre helyezhető *Emlékeztető*vel kapták meg.

A *Felhasználói kézikönyv* felhívja a használók figyelmét arra, hogy színes képet és hangot (fekete-fehér kép mellett is) csak kétnormás készüléken kapnak. Egyébként minden mai hazai gyártású tévé teljesíti e követelményt.

A C16 kiválasztásával egyidejűleg az Országos Pedagógiai Intézet irányításával megindult a kísérleti munka, egyelőre bizományi üzletekből vásárolt gépekkel. Többek között ennek eredményeként lehetett a megyei Pedagógiai Intézetek szakemberei részére ez év januárjában olyan tanfolyamot tartani, amely előkészítette az általuk tartandó megyei tanfolyamokat, és amelyen számos C16 segédprogram, bemutatóprogram és oktatóprogram cserélt gazdát. Ugyancsak ennek eredménye lett a Novotrade-nél megjelent *Feladatgyűjtemény C16 számítógéphez általános iskolásoknak*, illetve a *Tanári segédkönyv a C16 számítógéphez készült általános iskolai feladatgyűjteményhez* c. könyv. (Szerzőjük Bencsikné Takács Márta.) Az Országos Oktatástechnikai Központ *Mikroszámítógép-programok* sorozatának 3. kötete ugyancsak tartalmaz már C16-os programokat.

Az új elektronizációs oktatási program keretében meghirdetett hardverpályázat eredményeképpen — mint lapunk már közölte — az általános iskolai programban továbbra is számításba jön a C16, illetve a vele kompatíbilis 64 kbájtos Commodore Plus/4. A vállalkozó szállítók biztosítják a devizát, a Commodore felelős vezetői szerint gyártják a jövőben is e gépeket. Várható ugyan a nyugati piacokon a szoftverkínálat bővülése, de oktatásunkban — az eddigiekhez hasonlóan — saját szoftverfejlesztésre vagyunk utalva. Ehhez jó alapot ad a 3.5 verziójú, igen kiváló BASIC, valamint a befejezéshez közeledő FORTH-fejlesztés és a már megszületett oktatási programok. De továbbra is várjuk olvasóink ötleteit, amelyek megkönnyíthetik a programfejlesztéseket.

DR. VARGA ANDRÁS
Tudományszervezési és Informatikai Intézet

Az ismeretlen C16

A KERNAL rutinok

A C16 lelkivilágának ismertetését a KERNAL rutinok leírásával folytatom. Ezek a rutinok a gép operációs rendszere szempontjából alapvető feladatokat látnak el. Különleges tulajdonságuk, hogy a legtöbb Commodore gépen azonos módon lehet használni őket. Hívásuk előkészítése, belépési címük és működésük lényegi része a különböző géptípusokon megegyezik egymással.

A belépési pontok a memória $FF lapjának felső címein helyezkednek el. Itt közvetlen és közvetett ugróutasítások sorozatát találhatjuk. Ezeken keresztül jut a vezérlés azokra a helyekre, ahol a tényleges végrehajtás történik.

A C16-on lévő KERNAL rutinoknak most kb. 1/3-át ismertetem. Igyekeztem úgy válogatni, hogy az összefüggő rutinok lehetőleg ne szerpeljenek külön részben; egyedül a BSOUT-nál tettem kivételt.

Az egyes eljárások leírásánál eltérek az előző részben követett módszertől: nem adom meg külön, hogy az említett rutin mely címen hívható a C64 és VC20 gépeken, mert ez ebben az esetben felesleges. Néhány hivatkozott rendszerváltozó címével is adós maradok. Most nem fontosak, később majd pótolom őket.

RAMTAS $FF87 65415
Megvizsgálja a RAM-ot és beállítja az ezzel összefüggő rendszerváltozókat. (A VC20-on ez a KERNAL rutin hiányzik.)

RESTOR $FF8A 65418
Beállítja, illetve helyreállítja a $0312—$0331 (786—817) címtartományban lévő rendszervektorokat. A C64 és VC20 típusú gépeken ezek a vektorok a $0314—$0333 (788—819) címeken találhatók, azonos sorrendben.

VECTOR $FF8D 65421
A RESTOR által is kezelt vektorokról másolatot készít vagy a felhasználó által kívánt értékre állítja be azokat, a carry bit értékétől függően.

Hívása előtt egy 16 bájtos terület címét kell az X/Y regiszterpárban elhelyezni, úgy, hogy az X-be a cím alacsony helyérté-

kű bájtja, Y-ba a magas helyértékű bájt kerüljön. C=1 esetén a rutin a vektorok tartalmát az így meghatározott területre másolja, C=0 esetén pedig a fenti módon megadott címtől kezdődően a következő 16 bájt tartalmát a rendszervektorokba írja.

SCNKEY $FF9F 65439

A lenyomott billentyűhöz tartozó ASCII kódot a billentyűzetpufferbe küldi. Ha híváskor nincs lenyomott billentyű, akkor hatástalan. Általában a GETIN rutinnal célszerű együtt használni.

READST $FFB7 65463

Az I/O állapotszó tartalmát az A regiszterbe tölti. (Ez a változó — mindhárom géptípuson — az $90 címen található. Az állapotszó tartalma 0, ha az utoljára végrehajtott beviteli/kiviteli művelet végrehajtása hibátlan volt, illetve 64, ha az olvasási művelet során a rendszer fájl-vége jelet észlelt. Minden egyéb érték átviteli hibát jelent.

BSOUT $FFD2 65490

Az A regiszter tartalmát az aktív kiviteli csatornára küldi. Ennek részletezésére a következő alkalommal visszatérek. Alaphelyzetben az a karakter kerül a képernyőre, melynek ASCII kódja az A regiszterben van.

SETTIM $FFDB 65499

Beállítja a 0 lapon lévő rendszeróra (angolul Jiffy Clock) tartalmát. Ez 1/60 másodpercenként számlál egyet. Az óra 3 bájtos, beállítása előtt az A regiszterbe kell tenni a kívánt érték legmagasabb helyértékű bájtját, az X regiszterbe a középső, az Y-ba a legkisebb helyértékű bájtot.

RDTIM $FFDE 65502

Ez a rutin a rendszeróra aktuális tartalmát az A, X és Y regiszterekbe tölti, a SETTIM rutin leírásánál ismertetett sorrendben. A BASIC TI nevű fenntartott változójának az értékének megfelelő eredményt ad, eltérő formában.

STOP $FFE1 65505

Megvizsgálja, hogy a hívásakor a stop billentyű le van-e nyomva. Ha igen, akkor (a következő számban ismertetendő) CLRCH KERNAL rutin hívásával lezárja a megnyitott fájlokat és az állapotregiszter Z-bitjét 1-re állítja. Ha a stop billentyű nincs lenyomva, a Z-bit értéke 0 lesz.

A SYS utasítással vagy az USR függvénnyel hívott gépi kódú alprogramok futása alatt a stop billentyű lenyomása hatástalan szokott lenni. Ezt a C16-on a következő utasítások megfelelő helyre való beírásával kerülhetjük el:

```
...
JSR $FFE1
BEQ $867E
```

Megfelelő helyen olyan pontot értek, ahová a programrész futása során elég gyakran adódik a vezérlés.

GETIN $FFE4 65508

A billentyűzetpufferben található következő karaktert az A regiszterbe teszi, és a karakter eltávolításával aktualizálja a billentyűzetpuffert. Ha a puffer üres, az A regiszterbe 0-t visz.

Példa az SCNKEY és GETIN rutinok együttes használatára:

```
CIKLUS JSR $FF9F  ; SCNKEY
       JSR $FFE4  ; GETIN

       CMP #$00
       BEQ Ciklus
```

A programrész egy billentyű lenyomására vár. A lenyomott billentyűhöz tartozó ASCII kód az A regiszterbe kerül, a további feldolgozás során felhasználható.

UDTIM $FFEA 65514

Megnöveli a 0 lapon lévő rendszeróra értékét. A gép hardver megszakítási (interrupt) rendszere hatvanad másodpercenként hívja ezt a rutint. Használata csak akkor célszerű, ha a hardver megszakítási rendszert SEI utasítással kikapcsoltuk.

SCREEN $FFED 65517

Az X regiszterbe teszi a képernyő oszlopainak, az Y regiszterbe a soroknak a számát. Ez az egyes géptípusoknál állandó, nem veszi figyelembe a képernyőméret esetleges szoftver úton történő módosítását. Jelentősége a más géptípusokon is futtatható programok írásánál van, de ez — úgy tudom — nagyon ritka.

PLOT $FFFO 65520

A carry értékétől függően beállítja vagy lekérdezi a kurzor pozícióját. Beállításnál a carryt 0-ra állítjuk (CLC-vel), az X regiszterbe visszük a kívánt kurzor sorának számát, az Y-ba az oszlop számát, azután hívhatjuk a PLOT rutint. Lekérdezésnél a carryt 1-re állítjuk, majd a PLOT hívása után az X/Y regiszterpárban megkapjuk a kurzor helyének sor/oszlop koordinátáit.

Fontos! A sorok és oszlopok számozása 0-tól kezdődik. Célszerű arra is ügyelni, hogy az X regiszterbe kerül a függőleges koordináta, az Y-ba a vízszintes, éppen ellenkezőleg, mint ahogyan azt a geometriában megszoktuk.

A PLOT rutint a C64 és VC20-as gépeken gyakran hívják BASIC-ből is a kurzor beállítására. A C16-nál ezt a CHAR utasítás feleslegessé teszi.

BARNA LÁSZLÓ

Önkritika a „Hardcopy C16-hoz” című cikkhez

A µM 1986. júniusi számában megjelent cikkben helytelen megállapítás volt, hogy a monitorszolgáltatások nyomtatásához külön hardcopy program szükséges. Természetesen ugyanolyan egyszerűen nyomtathatók mint a BASIC listák, a CMD utasítás alkalmazásával. Azért marad még elég eset, amikor szükség van a programra!

VARGA ANDRÁS

Helyreigazítás

A lapunk 1986/5. számának 24—25. oldalán közölt µ'86 riportban durva hibát követtünk el. A cikk írója Kozma Lászlónéval készített interjút és nem Kalmár Lászlónéval, ahogy riportjában írta. Gondatlanok voltunk, nem tudjuk megmagyarázni, hogyan történt. Kozma Lászlónétól és Kalmár Lászlónétól is elnézést kérünk.

A szerkesztőség

Felhívjuk kedves érdeklődő Olvasóink figyelmét, hogy az 1986/6-os (júliusi) számunkban a 21. oldalon közölt ATARI Szekcióval kapcsolatos helyes telefonszám:

128-266

(Hirschler Tamás)

Tájékoztatjuk kedves megrendelőinket, hogy az oktatási programok forgalmazását 1986. szeptember 1-jétől ismét intézetünk végzi.

Jelenleg a HT 1080Z számítógépekre készített oktatóprogramok széles választékával állunk az érdeklődők rendelkezésére.

Címünk: 1372. Budapest, Postafiók 454.
XI., Egri J. u. 1—9. „E” ép.
Telefon: 453-197

Tudományszervezési és Informatikai Intézet

Az 1986. évi „Nemes Tihamér” középiskolás számítástechnikai versenyről

Mint azt valószínűleg többen is tudják, a Neumann Társaság kezdeményezésére, más intézményekkel együttműködve, már 1980 óta szervezünk különféle számítástechnikai versenyeket középiskolások részére. Tavaly kapta meg a számítástechnika azt a jogot, melyet az iskolai szaktárgyak országos tanulmányi versenyeinek győztesei már régebb óta élveznek: az első tíz helyezettet mentesítik matematikából az egyetemi, főiskolai felvételi alól. Ugyancsak tavaly, az országos matematikai versenyek mintájára neveztük el versenyünket a kibernetika egyik legelső kiemelkedő hazai művelőjéről, terjesztőjéről, Nemes Tihamérról.

A számítástechnikai versenyeztetésnek immár ötéves hagyománya, hogy kétfordulós, mégpedig oly módon, hogy az első fordulóban számítógép használata nélkül teszszük próbára versenyzőinket. Ennek egyszerű magyarázata az a kényszer, hogy napjainkig még nem volt meg a reális lehetősége annak, hogy az országban egyazon időpontban géphez férjen 2-3000 tanuló. A második fordulóban azonban múlhatatlanul szükségesnek tartottuk a gépen történő programfejlesztési képességek lemérését. Emiatt — amióta ez a lehetőség megvan — évről évre egy helyen összeszedünk nagyobb számú mikroszámítógépet (mégpedig kizárólag HT—1080, HT—2080 és Commodore 64 típusúakat, mivel a középiskolákban ezek terjedtek el), és a forduló tétje adott feladatra program készítése. A gépek számának és a helyiségek férőhelyeinek korlátai természetesen egyben a második fordulóba juttatható versenyzők számának korlátait is jelentik. Tavaly például 50, idén 73 második fordulós versenyeztetését tudtuk megoldani.

Mielőtt az idei verseny körülményeiről, feladatairól szólnánk, néhány gondolatot felvázolunk a verseny céljairól, feladatairól. Ezek a célok összetettek, és a teljesség igénye nélkül ragadunk ki néhányat, melyet jelenleg már látunk és viszonylag jól meg tudunk fogalmazni.

— Közismert, hogy a számítástechnika iránt a tanulóknak egy nem elhanyagolható része spontán érdeklődik, szinte szenvedélyel foglalkozik vele. Ennek a belülről jött motivációnak azonban megvannak a maga veszélyei is. Egyebek között például az, hogy a tanulók a számítógép használatának csak bizonyos szűk területeit veszik észre, csak ilyenekben merülnek el, emiatt érdeklődésük sőt készségük is, szakmai értelemben beszűkül. (Például csak játékok készítésére használják a gépet, csak a gépi kód és tármanipuláció trükkjeiben gondolkodnak, és távol maradnak a számítástechnika tényleges gyakorlati hasznosíthatóságának problémaköreitől.) *A verseny egyik feladata bemutatni azokat a követelményeket, irányzatokat, amelyek a korszerű, jövőbe tekintő számítógép-alkalmazások területén mutatkoznak.* Természetesen figyelembe véve az életkori és egyéb objektív korlátokat, leszűkített, leegyszerűsített feladatokról van szó (csakúgy, mint például a fizikai példák esetében, ahol számos olyan zavaró tényezőtől eltekintünk, melyeket valóságos számításoknál nem szabad elhanyagolni).

— A számítógép, maga a programnyelv, a gépeken használt felhasználói programok nem egyebek mint eszközök, különféle, rendkívül széles felhasználási területeken. Ez a tanuló részére két fő konzekvenciával jár: egyfelől tudatosulnia kell annak, hogy sem a gép, sem a gépi programozás, sőt a gépi játékok és egyéb hasonló programrendszerek nem tekinthetők céloknak, a velük való foglalkozás *csak ideig-óráig maradhat puszta szórakozás.* A másik következmény, hogy ha a tanuló már szerzett bizonyos jártasságot a géphasználatban, valamilyen programnyelv kezelésében, akkor szinte akarva-nemakarva feladatává válik más (nem számítástechnikai) felhasználók részére alkalmazási lehetőségeket teremteni a számítógépen. Ez nem csupán technikai követelmény, hanem egyben nevelési feladatot is jelent: *ki kell alakulnia* az ilyen tanulókban a felhasználó iránti barátságosságnak, *annak a tulajdonságnak, amellyel elébe megy a felhasználó igényeinek,* törekszik részére a lehető legésszerűbb és legkényelmesebb számítógép-alkalmazási helyzet kialakítására. Mindez nemcsak a felhasználó problémáinak megértését jelenti, hanem e probléma rendszerének, algoritmusának meglátását is.

— Célja továbbá a versenynek a lemérés és vetélkedés, ami számítástechnikai vonatkozásban legalább kétféle szempontból érvényesül: 1. Le kell mérnünk azt a *technikai készséget,* ami a gép, a programnyelv, illetve egyéb *szoftvereszközök kezelésében* nyilvánul meg. Ez természetesen conditio sine qua non, de semmiképpen sem a döntő kritériuma a mérésnek. 2. Vizsgálat tárgyává kell tenni *azt a tapasztalatot, ötletességet, amivel a számítástechnikai eszközöket* a gyakorlatiasan megfogalmazott *alkalmazás, megvalósítás szolgálatába képes állítani.* Hangsúlyozom, hogy ez az alkalmazásra való irányultság a legmarkánsabb kritériuma a mérésnek, ezt értékeljük legtöbbre. (Más kérdés, hogy az ehhez megfelelő feladatok kitűzése középiskolások részére korántsem egyszerű dolog.)

— A versenynek — különös módon — van egy olyan célja is, mely a számítástechnikát tanító középiskolai tanárokra vonatkozik. Mi úgy látjuk, hogy bár igen sok, lényegesnek mondható lépés történt a számítástechnikát és alkalmazásait oktató tanárok képzése és továbbképzése terén, a mai helyzet e tekintetben még nem megnyugtató. A szakmai, szakdidaktikai területen a részükre nyújtott segítség kevés. Ennek okaival természetesen itt nem foglalkozhatunk, de a verseny lehetőséget ad egyfajta segítségre, amit mi célunknak is tekintünk. A versenyek feladatai mindkét fordulóban tájékoztatják, segítik a tanárokat abban, hogy melyek azok a szaktárgyi problémakörök, melyekre tanulóikat felkészítsék. Milyen színvonalat, milyen mély problémaátlátást, milyen bonyolultsági fokot követelhetnek meg a legjobb tanulóktól, milyen módszereket használjanak az ismeretek lemérésében, az egyes készségtípusokat milyen súllyal értékeljék stb. A verseny értékelése párosuljon egyfajta visszajelzéssel a tanár részére, mely akár egyetértést, akár vitát vált ki, csak hasznosnak járhat mind a tanári munka, mind a versenyeztetők felfogása terén. Mindez természetesen csakis akkor válhat valóra, ha mi magunk is olyan színvonalon és igényességgel tudjuk versenyfeladatainkat kitűzni, hogy az elmondott céloknak megfeleljenek. Úgy érezzük, ezen a területen még bőven akad tennivalónk.

Az elmondottakból világos, hogy kerülni szeretnénk az egyes gépekre szűkült ismereteket, az egyes gépek BASIC-jeinek speciális reprezentációira vonatkozó tudás mérését, elismerését. Még inkább kerülni kívánjuk a gépi kódos és assembly szintű programozási ismeretek használatát, értékelését, beleértve a PEEK és POKE utasításokkal történő tármanipulációkat, melyekről tudjuk, hogy ezekben szinte valamennyi gép alapvetően különbözik.

Az alkalmazás-orientáltság természetesen tartalmazza a részben szoftver, részben hardvertechnikát feltételező berendezésszabályozási, illetve vezérlési területeket is. Mi eddig ilyennel nem foglalkoztunk, de gondoljuk, hogy előbb-utóbb aktuálissá válik majd egy ilyen témakörben folyó versenyeztetés is, mely a megfelelő feltételrendszerben alakítja ki céljait, módszereit. Már ma is szép számmal akadnak „chip-bütykölő” diákok, akikkel egy ilyen jellegű verseny résztvevői már biztosítottnak látszanak. Ez a profil azonban nem azonos azzal, amit a „Nemes Tihamér” verseny felvállalt.

Az 1986. évi „Nemes Tihamér” Országos Középiskolai Számítástechnikai Tanulmányi Verseny (ez a hivatalos neve) megrendezésében három intézménynek volt meghatározó szerepe: a Neumann János Számítógéptudományi Társaságnak, a Művelődési Minisztériumnak és a KISZ KB-nek. A feladat nagyjából úgy oszlott meg, hogy a gazdasági alapokat és az iskolák eljárásának szabályozását az MM biztosította, a számítástechnikai szakmai munkát az

NJSZT végezte, az anyagok terjesztésében és az adminisztráció jelentős részében pedig a KISZ KB segítségét élvezhettük.

A szakmai munkát egy ún. bírálóbizottság végezte, melynek közvetlen irányításával az NJSZT Oktatási Szakosztálya volt megbízva. Mindazonáltal a bírálóbizottság meglehetős nagy szakmai munkája mellett sem mentesülhetett egy sor szervezési, adminisztrációs feladattól. Ezen felül idén a verseny egész szervezése késve indult az 1985 végén lezajlott társulati választások miatt, de egyes elmaradásokban magunk is ludasak voltunk, így a határidők betartása olykor rendkívüli erőfeszítéseket kívánt meg. Az elmondottak, meg nyitottságunk okán felsorolom a bírálóbizottság tagjait: Deák-Jahn Gábor BME, Hanák Péter BME, Hunyady István BME, Kőhegyi János ELTE, Mohai Tamás BME, Somogyi László Ped. Szaksz., Székely Jenő Budapesti Tanárképző Főiskola, Tagányi György BME, Zámbó Viktor BME és Zsakó László ELTE. A bizottság vezetője e sorok írója volt. Mint látható, a bizottságba olyan, a számítástechnikai szakmához és a középiskolai oktatáshoz, sőt a fiatalokhoz közel álló személyeket válogattunk össze, akik saját munkájukban, érdekeikben összeférnek a versenyeztetőktől elvárt pártatlansággal. Ez utóbbi az oka, hogy aktív középiskolai számítástechnikai tanár nem került a bizottságba, hiszen egyébként nagyon sok olyat ismerünk közülük, aki felette érdemes lenne erre a feladatra.

A szakmai munka első fázisa egy olyan anyag elkészítése, amely nagy vonalakban vázolja a verseny céljait, továbbá azokat az alkalmazási területeket (leszűkítésekkel, korlátokkal együtt), amelyekben a tanulóktól bizonyos felkészültséget kívánunk meg. Erre már ez évben is kísérletet tettünk, szétküldtünk egy ilyen tárgyú anyagot. Úgy gondoljuk azonban, hogy ez valószínűleg még több lépésben javítandó, finomítandó, sőt egyes részeiben módosítandó lesz.

Az első forduló feladatainak kitűzése céljából a bizottság tagjaitól feladatjavaslatokat kértünk. A benyújtott írások anyagokat sokszorosítottuk, és a teljes szériát minden tagnak újra szétküldtük. Természetesen mielőtt ezek az anyagok elkészültek volna, egyeztető, elvi szempontokat kialakító megbeszélést tartottunk, melyben tisztáztuk, hogy mit is kívánjunk meg ettől a teszt jellegű versenyfokozattól.

A követelmények részben szakmaiak voltak, mint például: ne legyenek a feladatok gépre, BASIC-reprezentációra irányultak (amennyire ez lehetséges), de ha már mégis elkerülhetetlen, akkor egyaránt legyenek értelmezhetők HT és C64 tipusokon. Legyenek a kérdések gondolkodtatóak, vagy legalábbis adjanak fel értelmezési problémákat általánosan használt terminusokra nézve.

A követelmények másik része technikai volt: legyen a teszt viszonylag gyorsan javítható, a javítás könnyen áttekinthető. Ennek fő oka, hogy korábbi szokásainkhoz híven, nemcsak a tanárok, hanem mi magunk is kijavítunk minden dolgozatot. Úgy gondoljuk, erről is nyíltan kell szólnunk. Mind a korábbi, mind az idei tapasztalatok azt bizonyították, hogy a tanári javításokban meglepően nagy százalékban fordulnak elő hibák, nagyobb részt elnézésekből kifolyólag. Az értékelés megbízhatósága elengedhetetlenné tette a második javítást. Hangsúlyozni szeretném azonban, hogy ez kizárólag az értékelés objektivitásának javítását szolgálta és fogja szolgálni; mi senkit sem minősítünk javítási hibái miatt, még kevésbé adunk ki ilyen információkat bárkiről. Az igazi az lenne — ezt mi is világosan látjuk —, ha a javítási differenciákról a tanárok részére visszajelzést adnánk, természetesen indokolással együtt. Annak ma egy nagyon egyszerű, de igen nyomós oka van, hogy nincs így: nem győzzük az adminisztrációt, sem idővel, sem energiával. Ha a jövőben jobban sikerül megszervezni a verseny adminisztrációját (aminek igen objektív feltételei is vannak), akkor erre is sor fog kerülni.

Következő szakmai megbeszélésünkön a benyújtott választékból — egyes feladatokat még újra módosítva — meghatároztuk az első forduló végleges feladatsorát. Ezt követte a feladatsor értékelési szempontjainak, a feladatok, illetve részfeladatok súlyozásának, a pontszámozásnak kialakítása. A maximális pontszám az első fordulóban idén 66 volt. Az iskolából történő továbbküldés feltétele az lett, hogy a tanuló elérje az összpontszám egyharmadát, és ezen belül legyen legalább egy hibátlan megoldása a nem szakkifejezést értelmező kérdések közül. Az elkészült feladatsort és javítási útmutatót mágneses adathordozóra rögzítettük, onnan nyomtatva küldtük szét a bizottság tagjainak további javításra, stilizálásra. A kifejezetten gondos munka ellenére két komolyabb hiba került az anyagba, egyik a feladatkitűzésbe, a másik a tanári javítási útmutatóba. Az e hibákból eredő tanulói, illetve tanári értékelési hibákat természetesen megfelelő módon figyelembe vettük az újraértékelésnél. Az első forduló feladatsorát itt nem közöljük, mivel ezt minden versenyző iskola megkapta. Amikor elkészült ez az anyag, úgy éreztük, meglehetősen nehéz feladatsort állítottunk össze, tartottunk attól, hogy a maximalizmus hibájába estünk. Mint később kiderült, ebben bizony tévedtünk.

Az első fordulóban összesen kb. 1750 tanuló vett részt (becsült adat, mert kérésünk ellenére az iskolák egy része nem jelezte vissza, hogy hány nem továbbított dolgozat készült el), ami mintegy 900 tanulóval kevesebb a tavalyi összlétszámnál. A versenyben 273 iskola vett részt, melyekből 47 volt budapesti.

A továbbküldött dolgozatok száma (tehát azoké, akiknek pontszáma a felső kétharmadba esett) 1031. Ezeket 257 iskolából (ezen belül 45 budapesti) küldték el. Külön felhívom a figyelmet a következő adatra: az 1031 tanuló közül mindössze 26 volt lány. A beküldött tanulói munkák adminisztratív beiktatása során láttuk, hogy azok az általunk vártnál lényegesen jobban sikerültek, igen sok a nagyon magas pontszámot elért tanuló (a tanári értékelést figyelembe vébe). Ezért úgy határoztunk, hogy a bírálóbizottság azokat a dolgozatokat javítja újra, amelyek a tanári értékelés alapján a pontszámokat tekintve a felső egyharmadba esnek, azaz 45 vagy annál magasabb az összpontszámuk.

A beküldött dolgozatokból 426 tanuló jutott a felső egyharmadba, közülük 168 budapesti és a 426-ból két lány tanuló akadt. Mindez indokolttá teszi, hogy a lány tanulók számítástechnikai motiváltságának problémájára még visszatérjünk. A tanári értékelések és az általunk végzett újrajavítások viszonyaira az első 196 dolgozat alapján a következőket mondhatjuk:

— a dolgozatok 34,7%-ánál saját értékelésünk egyezett a tanári értékeléssel,

— a dolgozatok 35,2%-ánál emeltük a tanár által megítélt pontszámot, átlagosan 2,55 pontértékkel (a legnagyobb emelés 10 pont volt),

— a dolgozatok 30,1%-ánál csökkentettük a tanár által megállapított pontszámot, átlag 2,37 pontértékkel (a legnagyobb csökkentés 7 pont volt).

Az újrajavított dolgozatok általunk megállapított pontszámát tekintettük mérvadónak. Ebből következett, hogy a tanulók pontszám szerinti rangsorolása is kissé más lett, mintha azt az eredeti tanári pontozás alapján készítettük volna el. A pontszám szerinti rangsorolási listát, kizárólag a tanulók nevével, iskola megjelölése nélkül, egy e célra készített számítógépes program állította elő.

Mindezek után kerülhetett sor arra, hogy megállapítsuk, kik juthatnak a döntőbe. Ehhez természetesen ismernünk kellett azokat a korlátokat, melyek idén megszabták a második fordulóban részt vevők számának felső határát. Elsősorban a Budapesti Műszaki Egyetem segítőkészségének köszönhetően, mintegy 40 db HT és 15 db C64 állt a versenyzők rendelkezésére, három e célra alkalmasan berendezett teremben, mindenütt lehetővé téve a programlisták kinyomtatását is. A döntőbe jutottak választhattak (még a döntő időpontját megelőzően, postai értesítés alapján), hogy a két géptípus közül melyiken kívánnak versenyezni. A termek úgy álltak rendelkezésünkre, hogy két turnusban bonyolíthattuk le a döntőt: turnusonként kb. 36 tanulóval. Ennek előnye volt, hogy ekkora létszám (végül is 73 tanuló) eddig még soha sem juthatott döntőbe hazánkban. Hátrány volt viszont, hogy az időbeli megkötöttség miatt egy turnusnak maximum két és fél óra állt rendelkezésére feladata elkészítéséhez.

Miután a pontszám szerinti rangsorolásban pontosan 72 olyan tanuló volt, aki 59 vagy annál magasabb összpontszámot ért el, őket soroltuk a döntőbe. De azért nem jártunk el mechanikusan, mert valamennyi 58 és 59 pontos dolgozatot újra elővett a bizottság, és kollektíve ellenőrizte, újraértékelte a második javítást is. Ennek során egyetlenegy tanuló volt, akit 58 pontról 59 pontra felértékeltünk. Így jutott végül is 73 tanuló a döntőbe. Közülük 34 volt budapesti, és egyáltalán nem volt közöttük lány. A turnusokat úgy állítottuk össze, hogy a 8 órakor kezdő csoportba a budapestieket, illetve a Budapest környékieket soroltuk, és a 11-kor kezdő csoportba kerültek a vidékiek.

Természetesen már jóval a döntő előtt megkezdtük a kiadandó feladat kidolgozását, megfogalmazását. A szakmai előkészítés fázisai közül ezt tartjuk a legnehezebbnek, legigényesebbnek. Mert túlmenően azon az előzőekben már kifejtett szempontokon, hogy ti. egy a gyakorlatból vett feladat kerüljön ide, számos egyéb szempontra, korlátra is ügyelnünk kellett. Mindenekelőtt arra, hogy ez a gyakorlati feladat ne követeljen meg olyan elméleti háttérismeretet, melyet nem várhatunk el. Legyen a feladat olyan, hogy több, jól elkülöníthető részfeladatot és ezzel együtt jellemző telje-

sítményszinteket tartalmazzon, hogy az értékelés egyértelmű legyen. Igényeljen a feladat bizonyos rendszerszemléletet, adjon lehetőséget logikai egységek meglátására és ezzel együtt egy modul szerkezetű program elkészítésére, illetve az ennek megfelelő programlista-dokumentálásra.

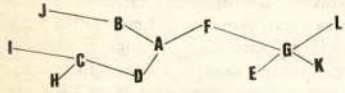
Mindebből nyilvánvalóan következik, hogy egy nagyobb méretű feladatot kell kitűznünk (és nem több kisebbet), de ez azért mégse legyen olyan nagy, hogy az adott időkeretben ne lehessen jól elkülönülő színvonalak elérése alapján értékelni. Úgy képzeltük, miután az első forduló feladatainak nehézségi fokát túlbecsültük, hogy ebben a második fordulóban az egymást követő részfeladatok közül az utolsónak valóban nehéznek kell lennie. Viszonylag hosszú kollektív bizottsági munka eredményeképpen készült el egy olyan feladat, melynek első része adatrendezésekből és listázásokból áll, második része egy fa-gráf útvonalon történő (nem optimalizáló) szállítási feladat, harmadik része egy csak néhány hurkot tartalmazó hálón történő (ugyancsak nem optimalizációs) feladat. A kitűzött feladat a következő volt:

Feladat: Tervezz és írj programot a Neumann János Számítógéptudományi Társaság tagjainak nyilvántartására, amely a számítástechnikai szakfolyóiratok szétküldésére és a szállítási útvonalak meghatározására szolgál.

Tárolandó adatok:

— név (max. 25 betű, ékezetes betűk nem szerepelnek)
— város (számuk max. 16, jelük egy betű az (A, P) tartományban)
— kért újság (vagy a Számítástechnika vagy a Mikroszámítógép Magazin rendelhető)

Az újságokat vasúton szállítják. A vasúti összeköttetéseket a felhasználó adja meg. A hálózat városokból, mint csomópontokból és ezeket összekötő vonalszakaszokból áll. A hálózat megadásának és tárolásának módját te határozod meg. Egy példa a lehetséges összeköttetésekre:



Megoldandó részfeladatok:

A) A tagok adatainak nyilvántartása:
1. Az adatbázis létrehozása legalább 30 tagra
2. Új tag felvétele
3. Tag törlése
4. Tag adatainak módosítása
5. A tagok névsor szerinti listája minden hozzá tartozó adattal
6. A tagok listája újságonként külön-külön, ezen belül városok szerint, ábécé sorrendben

B) Szállítással kapcsolatos feladatok:
1. A vasúti hálózat modelljének létrehozása a hozzá tartozó kezelőfunkciókkal. Ezek a funkciók a választott beviteli módtól függnek, így ezeket is neked kell meghatároznod.
2. Újságonként külön-külön lista készítése arról, hogy melyik városba mennyi újságot kell küldeni, és ebből hányat kell továbbítani a következő állomás(ok)ra. Feltételezzük, hogy mindkét újságot ugyanott, az „A" városban nyomtatják, és hogy „A" városból minden más városba el lehet jutni egy és csakis egy útvonalon.
3. A feladat ugyanaz, mint az előző pontban, de a két újságot bármelyik két különböző városban nyomtathatják, és néhány város több úton is elérhető.

Kiegészítések:

— A listákat képernyőre kell írni.
— Az adatbázisok háttértárolását nem kell megoldani.
— A program a lehető legjobban legyen dokumentálva, feleljen meg az általános formai követelményeknek, és legyen barátságos a felhasználóval.
— A program első sora egy REM sor legyen, és tartalmazza a következő adatokat: név, iskola, helység.
— A programodat két példányban rögzítsd.
— Beadandó a kazetta vagy lemez és minden, a könnyebb megértést segítő dokumentáció, vázlat.

A feladat kitűzésekor az alábbi elképzeléseink voltak:

— az A1—A6 részfeladatok rutin jellegűek, ezeket mindenki meg fogja oldani.
— Igazi próbatételt, ami a mezőnyt széthúzza, a B1 és B2 feladatok fognak jelenteni. Előnyük lesz azoknak, akik valamit olvastak már gráfokról, de azok is meg fogják tudni oldani, akik most találkoznak először a témával.
— A B3 feladatot csak a legjobbak lesznek képesek megkezdeni, befejezésére nem számíthatunk.
— Háttértárolás a géptípusok erős különbözősége miatt nem kívánható meg.
— A dokumentáltságon elsősorban a magyarázó, illetve a programot tagoló REM sorok használatát értjük, és természetesen nem kérünk valamiféle kezelési leírást mellékletként.

A javításhoz egy meglehetősen részletes értékelő pontozási rendszert készítettünk, melyben sokféle válaszlehetőséggel számoltunk. Pontozási rendszerünk részfeladatonként és bizonyos minőségi tulajdonságonként az alábbiak szerint alakult ki:

A1-re maximum 2 pont
A2-re maximum 4 pont
A3-ra maximum 4 pont
A4-re maximum 5 pont
A5-re maximum 7 pont
A6-ra maximum 5 pont
B1-re maximum 5 pont
B2-re maximum 15 pont
B3-ra maximum 15 pont
Programszerkezet maximum 5 pont
Barátságosság maximum 5 pont
Adat- és programbiztonság maximum 5 pont

A program szerkezetén a program moduláltsága, tagoltsága, a programlista olvashatósága, áttekinthetősége értendő. A barátságosság a felhasználóval való kapcsolat minőségét jelenti, a menük kezelhetőségét, az egyes részfeladatokból történő továbblépés képernyőn való dokumentálását, a program befejezhetőségét stb. A biztonság részben az adatokra vonatkozik, amennyiben védi a programot a feladat kitűzése szempontjából abszurd adatok bevitelétől, illetve védi a programot olyan beavatkozásoktól, amelyekre az gépi hiba jelzéssel megállhatna.

Mint utólag rájöttünk, ezzel a feladattal viszont kissé túllőttünk a célon, aminek azonban részben az is oka volt, hogy a feladat méretéhez mérten a rendelkezésre álló idő rövid volt. A döntő oknak mégsem ezt tartjuk, hanem azt, hogy szó sem volt arról, hogy az A1-től A6-ig terjedő feladatok rutinjellegűek lettek volna, ellenkezőleg, a többségnek már ez is gondot okozott. A B3-ba senki bele sem fogott, sőt a B2-t sem fejezte be senki. A legjobbak azonban — kb. a döntőbe jutottak 25%-a — valamilyen formában tervet készítettek a B2-re, és a legjobb 10 ennek egy részét programban realizálta. Itt tehát az időfaktor nagyobb súllyal esett latba. Ha figyelembe vesszük, hogy a feladatok összpontszámából a B3 pontjait minden további nélkül elhagyhatjuk, akkor 62 maximális pontszámmal számolva (és a feladat túlméretezettsége miatt ezt kell reálisnak tartanunk) az első tíz helyezett összpontszámáról és relatív készenléti fokairól azt mondhatjuk, hogy azok 41-től 36-ig terjedtek, azaz 66%-tól 58%-ig értékelt minőséggel sikerültek. Hozzá kell azonban mindehhez tenni, hogy még a legjobbak pontszámaiból is hiányzott az utolsó három minőségi értékelés számos — nem túl nagy ráfordítással elérhető — pontszáma.

Az első tíz helyezett:

1. Erdei Zsolt
Árpád Gimnázium, Budapest
2. Huszár Péter
Széchenyi Gimnázium, Sopron
3. Paller Gábor
Teleki Blanka Gimnázium, Budapest
4. Mazán Zsolt
Szilády Áron Gimnázium, Kiskunhalas
5. Makay Géza
JATE Ságvári Gyakorló Gimnázium, Szeged
6. Bélteky Zsolt
Herman Ottó Gimnázium, Miskolc
7. Jánossy Zoltán
Lovassy L. Gimnázium, Veszprém
8. Novák István
Berze—Nagy Gimnázium, Gyöngyös
9. Mamrovics László
Árpád Gimnázium, Budapest
10. Pallagi László
Vörösmarty Gimnázium, Érd

A felsoroltakon kívül két NJSZT különdíjat és három KISZ különdíjat osztottunk ki. A különdíjasok közül kiemelem Habony Zsolt (Földes Gimnázium, Miskolc) munkáját, aki a B2 feladat megoldásában egyedül alkalmazott igen szellemesen egy rekurzív módszert.

Nem kívánunk foglalkozni itt a verseny szervezésének, adminisztrálásának problémáival, noha bőven volna erről is mit írni. Pusztán annyit, hogy akik neheztelnek ránk az ebbeli fogyatékosságokért, azok joggal éreznek így, és tőlük ezúton is elnézést kérünk. Egyetlen mentségünk, hogy igen fárasztó ráfordítások ellenére sem értünk el e területen átütő sikereket. Többek között ebben is van még mit javítani munkánkon.

A verseny szakmai, szervezési tapasztalatai igen sok, most még végig sem gondolt konzekvenciával szolgáltak. Kettőről azonban mégis említést kell tennem.

— A tanulók iskolai és autodidakta számítástechnikai felkészülése ma ott tart, hogy a legjobbak meglepően magas prog-

ramutasítás-használati, programozástechnológiai színvonalra jutottak el. Nem keveseknél határozottan felismerni szakkönyvek ismeretét, egyfajta olvasottságot. Mindennek az az eredménye, hogy jól ismerik a számítástechnika szakkifejezéseit, értelmezésüket is beleértve, magas szinten kezelik azokat az építőelemeket, melyekkel egy professzionálisnak minősíthető programozó is dolgozik. Ugyanakkor azonban ezek egymástól elszigetelt, alig-alig összekapcsolódó ismereteket jelentenek akkor, ha a feladatban a probléma lényegét kell meglátni, ha tudatosan végig kell járni az algoritmuskeresés útját. A tanulók jelentős része előbb kezd programot írni, és csak azután — amikor a megírt program valamilyen szempontból hibásnak bizonyul — kezd a rendszeren gondolkodni. Maga a rendszerátlátás is gyenge oldaluk. Könnyű lenne itt ítéletet mondani, de ez a szemlélet sem tanulóban, sem tanárban nem alakulhat ki pusztán attól, hogy sokat gyakorol a gépen, és megtanulja a BASIC nyelv használatát. Ma hazánkban a legtöbb olyan számítástechnikai szakember, aki ismeri az alkalmazás-orientált programkészítés szakterületét a gyakorlatból, saját szakmai tapasztalatain keresztül fejlődött azzá, ami. Olyan szakkönyv, amely középiskolás szinten a fő eljárások, módszerek áttekintésével juttatná el használóját egyfajta „általános felhasználói programkészítési kultúrá"-hoz, ma még nemigen akad. A célt mégis fontosnak, az irányában való haladást a szakmai képzés meglehetős széles körében elkerülhetetlennek tartjuk. A szellemi tartalék ma a problémák mögött rejlő rendszerek átlátásában és a hozzá kapcsolódó alkalmazási algoritmusok megtalálásában, felfedezésében rejlik. Erre kell nevelnünk.

— Végül pedig törlesztem adósságomat a lány tanulók számítástechnikai motiválásának problémájáról kialakult véleményem vázolásával. A félreértések elkerülése érdekében előrebocsátom, hogy nem elsősorban a női egyenjogúságról kialakult nézeteim miatt tartom ezt a helyzetet súlyos bajnak. Hanem azért, mert teljesen nyilvánvaló, hogy az adminisztráció, banki, kereskedelmi stb. munkák területén néhány éven belül nem lesz olyan munkakör, amelyben ne kellene számítógépet kezelni, használni. Persze az ehhez szükséges ismeretek sem lesznek azonos színvonalúak, hiszen például egy szövegkezelő rendszer használatát jól meg lehet tanulni a BASIC programozásban való teljes tudatlansággal is. De mekkora előnyben lesznek azok az adminisztrátorok, pénzügyi, kereskedelmi, államigazgatási előadók, akik valamit tudnak is arról, amit nap mint nap használnak. Az élet sorozatban produkál olyan váratlan igényeket, melyekre a pusztán billentyűhasználatra megtanított ügyviteli alkalmazottak képtelenek lesznek reagálni. Az alkalmazkodás csak azoknak lesz birtokában, akik egy kissé belelátnak abba a rendszerbe, amit használnak. Mire lesznek képesek a számítástechnikai alkalmazások iránt közömbös, felkészületlen nők ebben a követelményáradatban? Mire lesz képes az iroda, hivatal, vállalat a megfelelően felkészült (és motivált) ügyviteli stb. dolgozók nélkül? Itt látom a veszélyt.

Befejezésül köszönjük a lelkes tanári munkát, a segítőkész iskolavezetési tevékenységet, ami nélkül ezek az eredmények nem jöhettek volna létre.

DR. ADA-WINTER PÉTER
és a bírálóbizottsági munkaközösség

HT—1080Z A lemezrendszer utasításai és függvényei

Aki már látta a lemezrendszer utasításkészletét, az tudja, hogy ezeket az utasításokat akkor sem fogadja el a gép, ha a rutincímet átírjuk. Például ha a CVI szót parancsként gépeljük be — azzal az álnok céllal, hogy így lépjen a gép monitormódba, hiszen a kezdőcímet átírtuk —, akkor éppen úgy ?SN ERROR-t kapunk, mintha át sem írtuk volna.

Próbáljuk meg azonban az A=CVI utasítást begépelni! Ha mindent ügyesen gépeltünk, akkor a gép monitor módba került. A magyarázat, gondolom természetes. A CVI nem utasítás, hanem függvény. Tehát alkalmazására ugyanazok a „törvények" vonatkoznak, amelyek például a SIN, COS, VARPTR stb. függvényekre. Törvényen azt az átfogó szintaktikai szabályt értem, hogy függvény sor elején nem állhat; vagy az egyenlőségjel jobb oldalán, vagy egy print utasítás után, mint kiírandó elem kell hogy legyen.

Ezeknek az utasításoknak a típusát tartalmazza az alábbi táblázat.

Utasítás	A rutin kezdőcíme a RAM-ban	Típus
CVI	4152	C
FN	4155	B
DEF	4158	C
CVD	415B	A
EOF	415E	C
LOC	4161	C
LOF	4164	C
MKI$	4167	C
MKS$	416A	C
MKD$	416D	C
CMD	4173	A
TIME$	4176	B
OPEN	4179	A
FIELD	417C	A
GET	417F	A
PUT	4182	A
CLOSE	4185	A
LOAD	4188	A
MERGE	418B	A
NAME	418E	A
KILL	4191	A
&	4194	B
LSET	4197	A
RSET	419A	A
INSTR	419D	B
SAVE	41A0	A
LINE	41A3	A
USR	41A9	C

Jelmagyarázat A: parancs
B: függvény
C: kiértékelő függvény

A Madách Gimnázium
μklubja

DIÁK C-64 ROVAT

LEMEZOLDAL-JELÖLÉS

A 1541-es meghajtóval rendelkező felhasználók általában lemezeiket „kivágják", és a második oldalt is felhasználják programtárolásra.

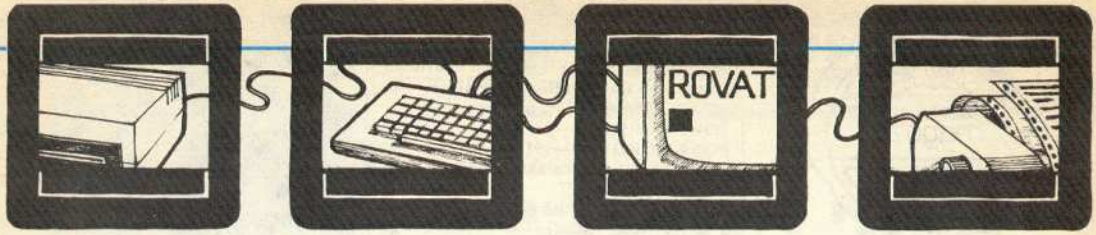
Hogy melyik a második oldal? Ez a számítógépeknek teljesen mindegy. Nekünk viszont tudni kell egy kívülről ráragasztott címkéről vagy a lemez más nevéből, ID-jéből. Az alábbi program felhasználásával mód nyílik egy olyan jelölésre is, amelynél a lemeznév és a lemez ID is ugyanaz maradhat mindkét oldalon.

Az ID után található egy 2A, a formatálás típusára utaló jelölés. Ennek B-re vagy A-ra (teljesen mindegy, lényeg, hogy két karaktert tartalmazzon) változtatása a lemezen lévő programokat nem érinti. Ez az átírás hajtható végre a program segítségével. Nekünk csak a jelölést kell beírni, s ahogy a lemezegységen a piros LED elalszik, a módosítás kész is. Az új tartalomjegyzék ellenőrzése előtt (LOAD"$",8) kapcsoljuk ki a gépet, ellenkező esetben nem érzékeli a változtatást.

A programot futtathatjuk akár olyan lemezen is, ami már foglalt, de vigyázni kell a pontos beírásra.

BÁRTFAI BARNABÁS
Ságvári Endre Szakközépiskola
Székesfehérvár

```
 0 REM ATOMSOFT B.B.
10 OPEN 1,8,2,"#":OPEN 15,8,15
20 S=18:SZ=0
30 P=165
40 INPUT"OLDALJEL";M$
50 PRINT#15,"B-R:";2;0,S;SZ
60 PRINT#15,"B-P:";2;P
70 PRINT#1,M$;
80 PRINT#15,"B-W:";2;0;S;SZ
90 CLOSE 1 : CLOSE 15
```



Amit nem tud a HT-val...

... azt mással kell megpróbálni. Természetesen a VAL-függvényről van szó. Az iskolaszámítógép kedvelői bizonyára irigykedve gondolnak a Spectrum, a ZX81 vagy az Aircomp (Homelab) remek VAL-függvényére. Ezek a gépek ugyanis képesek kiszámolni egy sztringben megadott kifejezés értékét, ami persze tetszőlegesen bonyolult lehet. Ha például az Aircompba beírjuk a PRINT VAL ("1+2") parancsot, akkor 3-at kapunk eredményül. Ha viszont a HT-nél kísérletezünk hasonlóval, a kiírt szám az 1 lesz, hiszen a gép csak a számokat értelmezi.

Az *1. listán* látható gépi kódú program ezt a hiányosságot orvosolja. Mivel magát a VAL-t kissé bonyolult lenne átírni, ezért a nem használt CVI-függvénynek adunk jelentést. A program a következő feladatokat látja el:

1. Átviszi a sztringet a 7F00H címre.
2. Tokenizálja/kódolja a műveleteket.
3. Kiszámolja és visszatér az aktuális értékkel.
4. Helyet foglal és átírja a CVI ugrótábláját.

Bár a program elfért volna az ugrótábla helyén is, egy ilyen kulcsszó használata „végzetes" lehetne. EDI-vel leforditva azonnal lehet használni a megújult VAL (CVI) függvényt. Írjuk tehát be a gépbe: PRINT VAL ("2*3"), CVI ("2*3"). Az eredmény 2 és 6 lesz. A különbség ugye óriási!

Hogy mire jó ez? Nos, vegyünk egy függvényábrázoló programot *(2. lista)*. Eddig a legjobb megoldást az önmagát változtató program kínálta. A dolgunk most lényegesen egyszerűbb. A függvényt, esetleg többet is, sztringként lehet tárolni, és értékét a változóktól függően bármikor kiszámíthatjuk. Vigyázzunk azonban, mert számolás közben ugyanúgy felléphetnek a különböző hibák (SN, FC, /0 stb. ERROR), tehát a BASIC programba be kell építeni a hibakezelést is. Ez az egyszerű ábrázoló a CVI lehetőségét és megkötéseit mutatja be, a bekért függvényt ábrázolja fél képernyőn. A CVI egyébként alig valamivel lassúbb egy direkt értékadásnál. A gépi kód betöltését és a helyfoglalást a 130-tól kezdődő rutin végzi, ezt kell tehát használni más programokban is.

PALLAGI LÁSZLÓ
Érd, Vörösmarty Mihály Gimnázium

```
 1 ORG 7ED0H
 2 LOAD 7ED0H
 3 CALL 2A07H
 4 JP Z,27F8H
 5 LD E,A
 6 INC HL
 7 LD C,(HL)
 8 INC HL
 9 LD B,(HL)
10 EX DE,HL
11 INC HL
12 LD DE,7F00H
13 PUSH DE
14 PUSH DE
15 PUSH DE
16 CALL 29CFH
17 EX DE,HL
18 LD (HL),0
19 POP DE
20 POP HL
21 CALL 1BCCH
22 POP HL
23 CALL 2337H
24 RET
25 ORG 4152H
26 LOAD 4152H
27 JP 7ED0H
28 ORG 40B1H
29 LOAD 40B1H
30 DW 7ECFH
31 END
```

1. lista

2. lista

```
10 GOTO130
20 CLS
30 PRINT@0,"F(X)=";:INPUTA$
40 CLS
50 ONERRORGOTO100
60 FORX=-30TO30
70 SET(X+31,23-CVI(A$))
80 NEXTX
90 GOTO30
100 E=ERR
110 IFE<>8ANDE<>10ANDE<>20THENRESUME20
120 RESUMENEXT
130 POKE16561,200:POKE16562,126
140 CLEAR1000
150 POKE16723,200:POKE16724,126
160 FORX=32464TO32498
170 READY:POKEX,Y
180 NEXTX:GOTO20
190 DATA205,7,42,202,248,39,95,35
200 DATA78,35,70,235,35,17,0,127,213
210 DATA213,213,205,207,41,235,54,0
220 DATA209,225,205,204,27,225,205
230 DATA55,35,201
```

A Spectrum rendszerváltozói

A Spectrum rendszerváltozók megtalálhatók ugyan a gépkönyvben, de a kétsoros magyarázatokból általában nem lehet tökéletesen kikövetkeztetni a változó szerepét. Ezen a problémán igyekszem segíteni a teljesség igénye nélkül, mivel egyes változók csak a rendszer számára értékesek, a programozó szempontjából nincsen gyakorlati hasznuk. Ezeket az interpreter használja, megváltoztatásuk a rendszer összeomlását idézi elő.

Az általános kényelmi szempontokat kiszolgáló rendszerváltozókat ismertetem. A táblázatban a cím hexadecimális formátumban is szerepel, mivel így gépi kódban könnyebben címezhető.

HEX	DEC	NÉV	FUNKCIÓ
5C08	23560	LAST-K	A legutoljára lenyomott billentyű kódját tartalmazza ASCII kódban. Kiolvasása előtt a FLAGS változó 5. bitjét kell ellenőrizni. Ha ez 1, akkor a LAST-K-ban érvényes új kód van. Kiolvasása után a FLAGS 5. bitjét 0-ba kell állítani.
5C09	23651	REPDEL	A változó azt az időt tartalmazza, ameddig nyomva kell tartani egy billentyűt ahhoz, hogy az ismétlési funkció (repeat) megkezdődjön. Ez az idő 1/50 másodpercben értendő. Kezdeti értéke 35. Gyakorlottabb gépelőknek érdemes kisebbre változtatni, így gyorsabb lesz az adatbevitel.
5C0A	23562	REPPER	Azt az időt tartalmazza, ami az ismétlési funkcióban két billentyűismétlés között eltelik. Kezdeti értéke 5. Megváltoztatásával szintén gyorsabb adatbevitel valósítható meg.
5C36	23606	CHARS	A karaktergenerátor kezdőcímét tartalmazza. Új karakterkészlet kialakításához ezt a címet a RAM-ba kell átírni, és innentől lehet az új karaktereket definiálni. Ezek a karakterkészletnek csak egy része, a 32 (space) és a 128 (©) közöttiek.
5C38	23608	RASP	A figyelmeztető zümmögés magassága, amit túl hosszú sor végén ad a gép. Átírásával a jelenleginél kellemesebb hangot is el lehet érni.
5C39	23609	PIP	A billentyű lenyomásakor megszólaló hang magassága. Szintén kellemesebbé lehet tenni nagyobb értékek beadásával.

RUSZNYÁK GÁBOR

DIÁK magazin ROVAT

HIBAIGAZÍTÁS

Lapunk 86/4. számában sajnálatos módon a 34. oldalon a Nagyító program ismertetése után egy ZX81-re írt Nagy felbontású grafika program listája jelent meg. Most közreadjuk a program leírását.

A 35. oldalon levő Matematika program leírása mellett a 34. oldalon ismertetett Nagyító program listája található. A Matematika program listáját most közreadjuk.

Nagy felbontású grafika

A program (listáját lásd a 86/4. szám 34. oldalán) csak közelítő pontosságú ábrát ad, ez azt jelenti, hogy egy bájtban a 256 variációból csak valamivel kevesebb, mint a fele jelenhet meg. A program 4 sorral nagyobb képernyőt használ a szokásosnál, azaz 28 sorosat. A felső 4 sor mindig az eredeti kis felbontású displayfájl első 4 sora. A legalsó 4 sor pedig mindig nagy felbontású. E két érték között szabadon állítható be a két képernyő határa. A nagy felbontású displayfájl 256 × 192 pixel.

A programot a 4084 hexa címtől kezdve fordíttassuk be egy REM sorba. Ezután írjuk be a mellékelt BASIC programot, mely a változókat definálja. Aztán mentsük ki a programot magnóra. Visszatöltés előtt vegyük le a RAMTOP-ot 6680H-ra és NEW-val töröljük a memóriát. Legelőször a CLS utasítást használjuk. Ez hozza létre a nagy displayfájlt. Az utasítás jellegű változókat USR-rel az adat jellegűeket POKE-kal aktivizáljuk.

Például: RAND UST CLS
POKE CORDX,15

A DISPLAYB-vel térhetünk át a nagy képernyőre. Ekkor a felső 4 sor BASIC-ből kezelhető. A többi nagy felbontású. Ettől kezdve a gép a hibajelzést felmásolja a képernyő mindig látható tetejére. Ez nagyban könnyíti a programozást.

A képernyő arányait a DISPLAYC-vel változtathatjuk. Ez a CORDY adatot használja.

POKE CORDY, (4—24)
RAND USR DISPLAYC

Visszatérni az eredeti képernyőhöz DISPLAYA-val lehet. Gyakran előfordul ZX81-en, hogy hibás POKE-olás miatt szétesik a displayfájl. Ha még él a BASIC, akkor ez a DISPLAYD rutinnal kijavítható a képernyő törlése nélkül. A PRINT rutinnal egy bájtot vihetünk a képernyőre.

POKE CORDX, (0—31)
POKE CORDY, (0—191)
POKE CODEA, a karakter kódja
RAND USR PRINT

A PLOT rutin közelítő pontossággal rajzol a képernyőre. Szintén a CORDX—Y-t használja.

DRAW-val vonalat húzhatunk. Változói CORDX—Y,
DRAWX—Y a vektor koordinátái.

Található még a programban scroll mind a négy irányban:
UP, DOWN, RIGHT, LEFT.

Gyors QSAVE és QOLAD, mely a nagy felbontású képernyőt kezeli.

INVERT utasítás és EXX rutin, amely minden A kódú karaktert B kódúra cserél. Formája:
POKE CODEA,.....
POKE CODEB,.....
RAND USR EXX

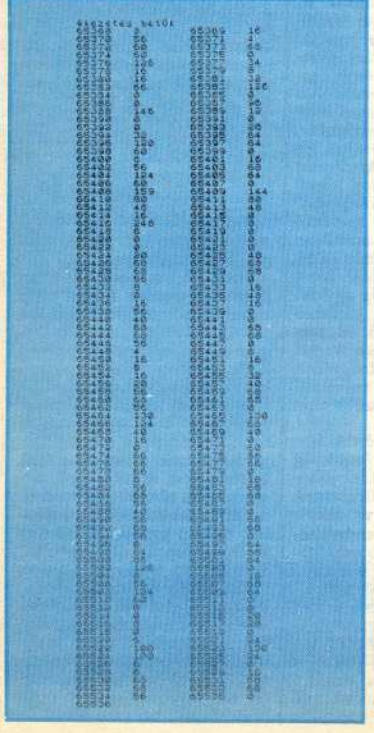
Végül a BORDER-t említeném, ami 3 pixel vastag keretet rajzol a nagy felbontású képernyőre.

PAPP ZOLTÁN

MATEMATIKA
Házi feladatok megoldására szolgáló program.

```
  1 POKE 23609,80: LET mb=3500
  2 LOAD ""CODE
  10 LET a$="By Papp Zoltán © 19
85"
  20 LET p=0: LET b=0
  30 DIM f$(6,64)
  40 FOR x=1 TO 6: LET f$(x)="X"
: NEXT x
  50 DIM z$(mb): DIM w(1): LET w
(1)=65
  60 LET z$(1 TO 21)=a$
  100 BORDER 6: PAPER 6: INK 0: B
RIGHT 0: OVER 0: CLS
  110 GO SUB 9000
  120 PRINT TAB 11; BRIGHT 1;"Mat
ematika"
  130 PRINT
  140 PRINT TAB 5;a$
  150 PRINT : PRINT TAB 6;"Kalkul
átor"
  160 INPUT "Válassz ! ";a
  165 IF a<1 OR a>5 OR a>INT a TH
EN GO TO 160
  170 GO SUB a*1000
  180 GO TO 110
  1000 GO SUB 9050
  1010 GO SUB 9010
  1020 PRINT TAB 1; BRIGHT 1;"Kalk
ulátor.
  1030 IF p=1 THEN LPRINT " Kalkul
átor.";CHR$ 13+CHR$ 13+a$+CHR$ 1
3
  1040 PRINT : PRINT "b=előző ered
mény, ret=vissza"
  1050 PRINT
  1060 INPUT c$
  1065 IF c$="" THEN GO TO 1060
  1070 IF c$="ret" OR c$="r" OR c$
="R" THEN RETURN
  1080 FOR x=1 TO LEN c$: IF c$(x)
>"(" AND c$(x)<")" THEN GO TO
1110
  1090 IF c$(x)>="RND" AND c$(x)<=
")" THEN GO TO 1110
  1100 IF "a"<=c$(x) AND c$(x)<="f
" OR c$(x)="." THEN GO TO 1110
  1105 GO TO 1060
  1110 NEXT x
  1200 LET b=VAL c$
  1210 PRINT c$;"=";b
  1220 IF p=1 THEN LPRINT c$;"=";b
  1230 GO TO 1060
  2000 GO SUB 9010
  2010 PRINT TAB 1; BRIGHT 1;"Máso
dfokú egyenlet.+x^2+b+x+c=
0"
  2030 INPUT "a,b,c=? ";a1;",";b1;
",";c1
  2040 PRINT : PRINT "(";a1;")+x^2
+(";b1;")+x+(";c1;")=0"
  2050 LET d1=b1+b1-4+a1+c1
  2060 IF d1<0 THEN PRINT : PRINT
"Nincs megoldás.": RETURN
  2070 IF d1=0 THEN PRINT : PRINT
"Egy megoldás.": PRINT : PRINT "
X=";-b1/(2+a1): RETURN
  2080 PRINT : PRINT "X1=";(-b1+SQ
R d1)/(2+a1);TAB 15;"X2=";(-b1-S
QR d1)/(2+a1): RETURN
  3000 GO SUB 9010
  3010 PRINT TAB 1; BRIGHT 1;"Függ
vény definiálás"
  3020 INPUT "Függvény neve ? (a-f
) ";d$
  3030 IF d$<"a" OR d$>"f" THEN GO
TO 3020
  3040 LET c=CODE d$-96
  3045 LET g$="FN "+d$+"(x)="
  3050 PRINT : PRINT g$+f$(c)
  3060 INPUT "Változtatás ? (I/N)
";e$
  3070 IF e$<>"i" AND e$<>"I" THEN
RETURN
  3080 INPUT "FN (x)= ";f$(c)
  3090 PRINT "Helyett": PRINT g$+f
$(c)
  3100 GO TO c*100+3100
  3200 DEF FN a(x)=VAL f$(1)
  3210 RETURN
  3300 DEF FN b(x)=VAL f$(2)
  3310 RETURN
  3400 DEF FN c(x)=VAL f$(3)
  3410 RETURN
  3500 DEF FN d(x)=VAL f$(4)
  3510 RETURN
  3600 DEF FN e(x)=VAL f$(5)
  3610 RETURN
  3700 DEF FN f(x)=VAL f$(6)
  3710 RETURN
  4000 GO SUB 9010
  4005 LET p=0
  4010 PRINT TAB 1; BRIGHT 1;"Függ
vény ábrázolás."
  4020 INPUT "Origó ? (0-250,0-170
) ";ox;",";oy
  4030 IF ox>250 OR ox<0 OR oy<0 O
R oy>170 THEN GO SUB 1010: GO TO
4020
  4040 PRINT : PRINT "(";ox;",";oy
")"
  4050 INPUT "X(max) ? ";xm
  4055 IF xm<=0 THEN GO SUB 1010:
GO TO 4050
  4060 PRINT "Xmax=";xm
  4070 LET q=xm/(255-ox)
  4100 PAUSE 30
  4110 CLS
  4120 PRINT AT 0,INT (ox/8)-1;"Y=
";INT ((175-oy)*q*100)/100
  4125 LET n$=STR$ (INT (xm*100)/1
00)
  4130 PRINT AT 20-INT (oy/8),30-L
EN n$;"X=";n$
  4140 PLOT 0,oy: DRAW 255,0
  4150 PLOT ox,0: DRAW 0,175
  4152 LET rs=10+INT (LN q/LN 10+2
)
  4155 FOR r=INT ((-xm*ox/(255-ox)
)/rs)*rs+rs TO xm STEP rs: PLOT
r/q+ox,oy+1: PLOT r/q+ox,oy-1: N
EXT r
  4158 FOR r=INT ((-xm*oy/(255-ox)
)/rs)*rs+rs TO xm+(175-oy)/(255-
ox)*xm STEP rs: PLOT ox-1,r/q+oy: P
LOT ox+1,r/q+oy: NEXT r
  4160 INPUT "f(x)=? ";i$
  4170 INPUT "Df:a? ";x0;" ";x1
  4180 IF x1<x0 OR x0>xm OR x1<-ox
*q THEN GOSUB 1010: GO TO 4110
  4182 IF x1>xm THEN LET x1=xm
  4184 IF x0<-ox+q THEN LET x0=-ox
*q
  4190 LET x2=x0/q+ox
  4200 LET s=10^-6
  4210 LET x=x0
  4220 LET y=VAL i$/q+oy
  4230 LET x2=x2+s
  4240 LET x=(x2-ox)*q: IF x>x1 TH
EN GO TO 4500
  4250 LET y=VAL i$
  4260 LET y2=y/q+oy
  4270 IF y2<0 OR y2>175 THEN LET
s=1: GO TO 4230
  4280 PLOT x2,y2
  4284 IF y2=x THEN LET s=1: GO TO
4230
  4285 LET s=-9/(ABS (y2-x)/s)
  4286 IF s>1 THEN LET s=1
  4287 LET x=y2
  4290 GO TO 4230
  4500 INPUT "Copy,Ismétlés,Függ.a
ssz,Ret.";j$
  4510 IF j$="C" OR j$="c" THEN CO
PY: GO TO 4500
  4520 IF j$="i" OR j$="I" THEN GO
TO 4160
  4525 IF j$="r" OR j$="R" THEN GO
TO 4600
  4530 CLS : RETURN
  4600 INPUT "Függ(egyes asszimptó
ta)=";as
  4604 IF ABS (as/q+ox-128)>128 TH
EN GO TO 4600
  4605 PLOT as/q+ox,0: DRAW 0,175
  4610 GO TO 4500
  5000 GO SUB 9000
  5010 PRINT TAB 1; BRIGHT 1;"Szöv
egszerkesztő"
  5020 GO SUB 9020
  5025 PRINT "*Load,*Save,*Memória
bővítés, *Kalk,*Előről,*Print
,*New,*Ret, *Fel,*egy Szöveg"
  5026 PRINT : GO TO 5500
  5030 INPUT y$
  5035 IF y$="" THEN LET y$=" "
  5036 IF y$(1)="*" THEN GO TO 530
0
  5040 FOR x=1 TO LEN y$
  5050 FOR y=x+31 TO x STEP -1
```

```
5055 IF y>LEN y$ THEN LET y=1+LE
N y$: GO TO 5060
5060 IF y$(y)=" " THEN GO TO 505
0
5070 NEXT y: GO TO 5030
5075 IF w(1)+y-x+1>LEN z$ THEN P
RINT "Megtelt!": GO TO 110
5080 LET z$(w(1) TO w(1)+y-x)=y$
(x TO y-1)+CHR$ 13: PRINT CHR$ 8
+z$(w(1) TO w(1)+y-x-1)+CHR$ 13+
";"
5090 LET w(1)=w(1)+y-x+1
5100 LET x=y
5110 NEXT x
5210 GO TO 5030
5300 IF LEN y$=1 THEN GO TO 5030
5310 IF y$(2)="L" OR y$(2)="l" T
HEN LOAD "" DATA w(): LOAD "" DA
TA z$(): GO TO 5500
5320 IF y$(2)="S" OR y$(2)="s" T
HEN INPUT "File név ";s$: SAVE
s$ DATA w(): SAVE s$ DATA z$():
GO TO 5030
5325 IF y$(2)="F" OR y$(2)="f" T
HEN GO TO 5600
5330 IF y$(2)="M" OR y$(2)="m" T
HEN INPUT "Hány képernyő ? ";mb:
 LET mb=mb+704: GO TO 50+(mb<350
0)+5330*(mb>=3500)
5340 IF y$(2)="K" OR y$(2)="k" T
HEN GO SUB 1010: GO TO 5600
5350 IF y$(2)="E" OR y$(2)="e" T
HEN GO TO 5500
5360 IF y$(2)="P" OR y$(2)="p" T
HEN LPRINT z$(1 TO w(1)-1): GO T
O 5030
5370 IF y$(2)="N" OR y$(2)="n" T
HEN GO TO 50
5380 IF y$(2)="R" OR y$(2)="r" T
HEN RETURN
5400 GO TO 5030
5500 GO SUB 9010: PRINT z$(1 TO w
(1)-2)+CHR$ 13+";": GO TO 5030
5600 LET w(1)=w(1)-2
5610 IF z$(w(1))=CHR$ 13 THEN GO
 TO 5630
5620 LET w(1)=w(1)-1: GO TO 5610
5630 LET w(1)=w(1)+1: GO TO 5500
9000 PRINT
9010 PRINT
9020 PRINT
9030 PRINT
9040 RETURN
9050 INPUT "Printer ? (I/N) ";b$
9060 LET p=0
9070 IF b$="I" OR b$="i" THEN LE
T p=1
9080 RETURN
9900 CLEAR : SAVE "Matek" LINE 1
: SAVE "Betuk"CODE USR "A",160
9999 FOR x=0 TO 7: INPUT a: POKE
 USR "T"+x,a: NEXT x
```



SOFTinvest

a FORTH nyelv kedvelőinek ajánlja

IBM PC/XT és azzal kompatíbilis gépekre

MicFORTH fejlesztőrendszerét

Elsősorban az ügyviteli adatfeldolgozási alkalmazásokat támogatja. Tartalmazza a FORTH—79, FIG—FORTH, MVP—FORTH és a STRING—FORTH szavait, továbbá screen-editort, lebegőpontos számábrázolást, FORTH adatbázis-kezelőt, hibakeresőt és javítóprogramokat (a program tetszés szerinti helyen megszakítható).

Commodore 64 számítógépre

MAX—FORTH 85 fejlesztőrendszerét

Segédeszköz programozáshoz és teszteléshez. A programozási időt harmadára csökkenti, a futást 50-szer gyorsabbá teszi, mint a BASIC-ben.

Új címünkön várjuk érdeklődését!

1137 Budapest, Kun Béla rakpart 8.
(Újpesti rakpart)
Telefon: 129-230

A MINŐSÉGÜGY KÖZÜGY

Minőség – minősítés – értékítélet

A minőségre vonatkozó ítélet mindig valaminek vagy valakinek a számára valamilyen szempontú megfelelőséget, *értékességet mér, értékességről informál, értékességet indikál.* Fontos kérdések tehát, hogy *kinek, minek számára, milyen szempontú, milyen célra való értékességről van szó, és ezt hogyan mérjük.*

Még ha ugyanazt bíráljuk is, más lehet az értékességre (minőségre) vonatkozó ítélet aszerint, hogy kinek, minek számára alkottuk, aszerint is, hogy milyen célra, milyen szempontokból alkottuk, és végül hogy hogyan, milyen eljárással alakítottuk ki azt. Ebből azonban következik az is, hogy az olyasféle ítéletek, hogy ez vagy az „jó minőségű", *hiányosak és ezért értelmetlenek,* nemritkán szándékosan *félrevezetők is.* A minőségre vonatkozó ítélet helyes szerkezete a következő. *„X minősége Y számára, Z célra történő alkalmazásra, felhasználásra stb. V értékű (mértékű, mérőszámú stb.)."*

A minőségre, értékességre vonatkozó ítéletet alkotni *kikerülhetetlen* és mindig mindenki sőt *minden létező által állandóan gyakorolt tevékenység,* minden létezőnek fő funkciója. Ennek fontossága — noha szembeötlő — *nem kapja meg a kellő hangsúlyt.* Az *értékességre* vonatkozó ítéletalkotást külön *tanítani kellene.* Az életben való navigációnk *sikere függ attól,* hogy milyen, milyen minőségű, a minőségre, értékességre vonatkozó ítéletalkotási tevékenységünk.

Hogyan alkossunk tehát helyesen ítéleteket a minőségre, az értékességre vonatkozóan? Általános érvényű megállapítások, egyszerű szabályok nyilván nem létezhetnek, hiszen túl sok jellemzőt kell figyelembe venni. (Ismételjük: például, hogy *kinek, minek számára? milyen célból? milyen célra?* stb.) A helyzet azonban nem reménytelen. Nagy, mindent magába foglaló szabály gyakorlati elérhetetlensége ellenére van sok kis lokális eljárás, ami jó, hasznos, kezelhető. Ezek azonban túl sokan vannak ahhoz, hogy elviselhető terjedelemben tárgyalhatók legyenek.

A legtanulságosabb az lesz, hogy néhány konkrét módszert mutatunk be.

Adott tehát a személy, a szervezet stb., aki számára készül az értékességi, a minőségre vonatkozó ítélet. Az is adott, hogy minek a minőségét kell elbírálnunk. (Lehet szó például Kovács úrról és tisztagyapjú öltönyökről egy konkrét áruház felsőruházati osztályán, vagy Kovács úr gyermekeiről és huszonötezer forint alatti, új személyi számítógépekről ugyanannak az áruháznak egy másik osztálya készletéből, de lehet szó Kovács úr által kidolgozott beruházási javaslatokról, amiket saját vállalata számára készített.)

A minősítendőket jellemzőikkel leírva ábrázolhatjuk egy olyan térben, amelyek koordinátatengelyein az egyes jellemzők skálái vannak. (Számítógépnél például az ár, a garanciális idő, a központi tár kapacitása bitekben stb.) Ezt a teret hívjuk *reprezentációs térnek.* Ennek pontjai (esetleg más alakzatai) egy-egy minősítendőt, illetve választási lehetőséget reprezentálnak, illetve reprezentálhatnak. Az egyik pontot (vagy más reprezentáló alakzatot) *jobbnak* ítélhetjük egy másiknál, ha a neki megfelelő választási lehetőség jobb mint a másik pontnak (alakzatnak) megfelelő. Így tehát pontok, alakzatok jóságáról, értékességéről, minőségéről is beszélhetünk. A minőség, értékesség tehát a reprezentáló pontokhoz, alakzatokhoz valamilyen módon hozzárendelt érték. Ilyen egyszerű az egész. Adott például egy alkalmasan szerkesztett többváltozós függvény, amelynek értelmezési tartománya egy többdimenziós tér vagy annak egy részhalmaza, értékkészlete pedig egy valós intervallum. A minőségre, értékességre vonatkozó ítélet kialakítása semmi másból nem áll, mint ilyen függvényekbe való helyettesítésből. E függvények azonban nem akármilyen függvények. Egy-egy ilyen függvény megfelel a már említett sok kis lokális szabály valamelyikének. Ezeket a függvényeket sajátságaik teszik alkalmassá arra, hogy konkrét helyzetekben alkalmazhatók legyenek. Az ilyen minősítő, értékelőszámot szolgáltató függvények fontossága, gyakorlati értéke tehát rendkívül nagy. E függvények az operációkutatás *„célfüggvényeinek" közeli rokonai.* (Egy jó „célfüggvény" kialakításának jelentősége pedig közismert.)

Lássuk tehát, hogyan készíthetünk például saját magunk számára, valamilyen célra *értékelő, minősítő függvényt!* A sok lehetőség közül bemutatunk egy egyszerű és könnyű gyakorlati utat.

Vegyük a lehetőségek között a lehető (reális) legjobbat és legrosszabbat! Egy lehetőséget a legjobb lehetőséghez és a legrosszabbhoz való *viszonya alapján minősítünk.* Ezt is *sokféleképp* tehetjük meg. Képezzük *például* a minősítő számot úgy, hogy a minősítendőnek a legrosszabb lehetőségtől vett valamilyen távolságából levonjuk a minősítendőnek a legjobb lehetőségtől vett valamilyen távolságát!

Természetesen több „rossz" lehetőséget és több „jó" lehetőséget is kijelölhetünk, és ezek mindegyikét egyszerre szerephez juttathatjuk az értékelő függvényben. Alkothatnak továbbá a kijelölt jó és rossz lehetőségek (esetleg pontonként mértékben és előjelben is különböző „jóságú" és „rosszaságú") tartományokat is. A bírált lehetőség (például pont) eltérését a rossz lehetőségek tartományától értelmezhetjük geometriailag, valamilyen távolsággal, de például integrálokkal, többek között potenciál jellegű integrálokkal is. Érdekes és gyakorlati szempontból nagyon nagy jelentőségű az, hogy az operációkutatás legtöbb célfüggvénye *nagyon egyszerűen származtatható az említett, geometriai jellegű, eltéréseket (távolságokat) használó megfontolásokkal.* Ez a tény — noha nem közismert — természetes: minden létező igyekszik saját tulajdonságainak, adottságainak, korlátainak megfelelően *(a) rossz lehetőségektől minél távolabb és (a) jó lehetőségekhez minél közelebb kerülni,* mégpedig egyidejűleg, valamilyen rá jellemző módon. Az értékelő függvények egymástól való szerkezeti és értékbeli különbségeiben mutatkoznak meg az egyes minősítők sajátságai közötti eltérések.

A minősítendők, értékelendők egyes jellemzői fontosság és értéktartomány-elhelyezkedés szempontjából eltérhetnek egymástól; ezeknek az eltéréseknek helyes figyelembevétele súlyozással is, koordinátatranszformációval is, az értékelő függvény megfelelő kialakításával is megoldható. E műveletek is függnek természetesen attól, hogy a minősítési, értékelési ítélet kinek, minek a számára, milyen célra készül.

Legyenek egy minősítendő jellemzői $j_1, \ldots j_n$, és minden jellemző olyan, hogy *nagyobb értéke a jobb.* Minősítendőnk tehát egy $(j_1, \ldots, j_n)$ vektorral, illetve egy J ponttal reprezentálható, $(J=(j_1, \ldots, j_n))$.

A J által reprezentált lehetőséget egyszerűség kedvéért pl. öt lehetőséghez viszonyítva fogjuk megítélni. Az öt lehetőségből három (U, V, Z) legyen rossz, kettő (A, B) pedig jó. Egy f értékelő függvényt például a következő módon lehet definiálni.

$f(J)=f(U, V, Z, A, B, J)$

Konkrétabban például: $f(J)=f(\rho_1(U, J), \rho_2(V, J), \rho_3(Z, J), \rho_4(A, J), \rho_5(B, J))$ ahol $\rho_1 \ldots, \rho_5$ valamilyen eltérés, például valamilyen távolság.

Ez az értékelőfüggvény-definiálási mód rendkívül egyszerű és numerikusan könnyen kezelhető, még olyan esetekben is, amikor A, B, . . ., U, V, . . ., illetve J nem pontok, hanem egyenesek, síkok, kör, gömb vagy más tartományok.

Nem nehéz belátni, hogy alkalmazható ez a módszer olyankor is, amikor nem egy értékelendőre, hanem értékelendők egy összességére vonatkozóan kell minősítő ítéletet kialakítani.

Egyszerűség kedvéért álljon n elemből a minősítendő csapat (pl. brigád) és legyen $k+l$ vonatkoztatási pontunk, k darab rossz, l darab jó. (A minősítendők $J_1, \ldots, J_n$ a rossz pontok $R_1, \ldots, R_k$, a jók $A_1, \ldots, A_l$.) Egy egyszerű és numerikusan könnyen kezelhető lehetőség a következő.

$$f(J_1 \ldots J_n)=\sum_{i=1}^{n}\sum_{j=1}^{k}\rho_{ij}(J_i, R_j)-\sum_{i=1}^{n}\sum_{j=1}^{l}\sigma_{ij}(J_i, A_j),$$

ahol a ρ_{ij} és a σ_{ij} valamilyen eltérést (távolságot) adó függvények.

Ha a „jó", a „rossz" és a minősítendő pontok száma megszámlálható végtelen, végtelen sorokkal kell dolgoznunk. Nagyobb számosságok esetében — ha az ilyen feladat numerikusan kezelhető —, integrálok kiszámítását (értékének megbecslését) kell elvégeznünk. Fontosnak tartjuk hangsúlyozni, hogy ezek, a látszólag csak matematikai érdekességű, bonyolultnak tűnő esetek nem elméleti spekulációkból születtek, hanem gyakorlati igények kielégítésére készült egyszerű eszközök az értékelés, a minősítés gazdag módszertárából.

Gyakorlásul írja fel az olvasó annak a minősítő függvénynek a formuláját, amely n elemű csapatokat minősít, k rossz és l (különböző) jó lehetőségtől vett távolságösszegek (eltérésösszegek) hányadosával!

Assemblerek, cross-assemblerek

7. Az operátorkészlet II. – A rendszer kapcsolatábrája

16. operátor: operandusmező-kiértékelő *(1. ábra)*

Feladata: a műveletikód-információk meghatározta módon értelmezett operandusmezőt értékeli ki.

Regiszter, regiszterpár és/vagy aritmetikai kifejezésekből felépített opreandus(ok) értékét számítja ki. Ha több operandusa van az utasításnak, akkor az egymástól vesszővel elválasztott operandusokat egyenként határozza meg. Regiszter és regiszterpár típusú operandus(ok) esetén a mnemonikus regiszternévhez tartozó és a tárgykódba beépítendő bináris adatot az ún. regiszter-, illetve regiszterpár-kódtábla tartalmazza.

Egyéb típusú operandus a következő egyszerű kifejezésekből épülhet fel: hexadecimális, oktális, bináris, decimális számkonstans; a beültetési számláló aktuális értéke; ASCII konstans (karakter típusú konstans); címke; azonosító.

Egy címke vagy azonosító értékét az aktuális címketáblából keresi ki az operátor, tehát ilyenkor aktivizálja a 8. operátort. Makrokifejtés esetén, ha az aktuális makrocímketáblában nem találja meg a keresett szimbólumot, akkor a modulcímketáblában folytatja a keresést.

A fenti egyszerű kifejezésekből a következő — csökkenő precedencia szerint megadott — aritmetikai és logikai műveletekkel alakítható ki egy összetett kifejezés: negáció, negatív előjel; szorzás, egész osztás, bitenkénti léptetés jobbra és balra; összeadás, kivonás; logikai szorzás; logikai összeadás, kizáró vagy művelet (NOR).

Az egy összetett kifejezésben szereplő műveletek és egyszerű operandusok száma korlátozott. A kifejezés értékének meghatározása az ismert inverz lengyel módszerrel történhet.

Az operátornak működése végén nemcsak az operandus(ok) értékét, hanem az operandusok számát is jeleznie kell, s utalnia kell arra is, hogy az operandusok értéke relatív vagy abszolút-e. (Ezeket az indikátorokat a 18. operátor használja fel.) Címkét és a beültetési számláló értékét tartalmazó kifejezések lehetnek relatív típusúak, azaz olyan kifejezések, amelyek értéke a fordítási kezdőcím értékétől függ.

Amennyiben az operátor az operandusmező kiértékelése során bármelyik operandusnál szintaktikai vagy szemantikai hibát fedez fel, akkor beállít egy hibaindikátort, és az operandusmező, illetve az operandus értékeként nullát ad ki.

Nyolcrészes sorozatunk a mikorszámítógépek assemblereiről, cross-assemblereiről szól. Célja, hogy a cross-assemblerek példáján keresztül megismertesse az olvasót az assembler programok működésével. A bemutatáshoz a rendszermodellezési eszközöket használjuk fel, s így készítjük el az assemblerek működésének egy szabványosított algoritmusát, modelljét. E cikkünk a sorozat hetedik része.

1. (Cross-)assemblerek és a rendszermodellezés
2. Az Intel 8080 assembly nyelv
3. A (cross-)assembler program mint rendszer
4. A rendszer működése I. — A fordítás két menete
5. A rendszer működése II. — Táblák és adatterületek
6. Az operátorkészlet I.
7. Az operátorkészlet II. — A rendszer kapcsolatábrája
8. Példák a rendszer működésére

17. operátor: tárgykód-meghatározó *(2. ábra)*

Feladata: a műveletikód-információk és az operandus(ok) értékének segítségével meghatározza a műveletikód-mezőben álló gépi utasítás tárfoglalásának hosszát (mindkét menetben) és tárgykódját (csak a második menetben). A tárgykódba beépíti az esetleges regiszter- vagy reiszterpár-kódot, illetve az egyéb típusú operandus értékét.

18. operátor: tárgyprogram-készítő *(3. ábra)*

Feladata: a korábban meghatározott tárgykódot kiegészíti a tárgykód flag- és hosszadatokkal, s ennek alapján bejegyzést készít a tárgyprogram pufferterületén (OBJB). Előzőleg az operátor ellenőrzi, hogy a kiszámított hosszúságú bejegyzés elfér-e a pufferben. Ha nem, akkor lezárja a tárgyprogram-rekordot és kiviszi a kívánt kimeneti perifériára, és a puffer mutatóját a terület elejére állítja be.

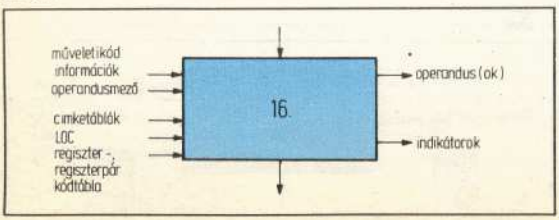
A tárgykód flagek egy lehetséges kiosztása a következő:

1: gépi utasítás abszolút típusú operandussal vagy operandus nélkül,
2: gépi utasítás relatív típusú operandussal,
3: ORG direktíva,
4: END operandus nélkül,
5: END abszolút típusú operandussal,
6: END relatív típusú operandussal,
7: DB (mindig abszolút),
8: DW abszolút típusú operandussal,
9: DW relatív típusú operandussal,
A: DS direktíva (mindig abszolút),
B: tárgyprogramrekord vége,
C: a tárgyprogram vége.

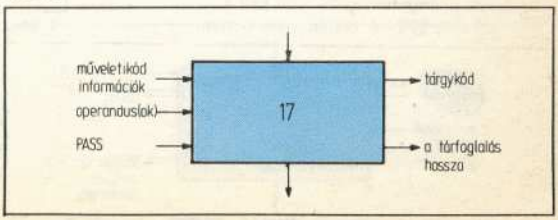
19. operátor: direktíva-feldolgozó *(4. ábra)*

Feladata: a fordítóprogramnak szóló utasítások (direktívák) végrehajtása. Így például az operátor működése során sor kerülhet bizonyos rendszerváltozók értékének megváltozására (IF,ENDIF,MACRO,ENDM,END direktívák esetén), a címketáblában készített egy bejegyzés módosítására (EQU,SET), a beültetési számlálónak történő értékadásra (ORG), a kívánt tárfoglalás hosszának és esetleg tartalmának meghatározása (DB,DW,DS), a forrásprogram végének érzékelésére (END), a

1. ábra



2. ábra



tárgyprogramba történő bejegyzés elkészítésére (ORG,END,DB,DW,DS), s bizonyos szabályozó-vezérlő indikátoroknak és hibaindikátoroknak a beállítására például a 18. operátor számára (valamennyi direktívánál).

Az elvégzendő műveletek nemcsak a direktíva típusától, hanem a fordítási menettől is függenek.

Minden direktíva feldolgozását végző programrészlet (ehhez mutat a műveletikód-információban elhelyezett ugrási cím) önálló, s működésében egymást kizáró operátorként fogható fel. Az ábra egyszerűsítése érdekében vontuk őket egyetlen operátorba össze.

Az egyes direktíváknál elvégzendő műveletek, azaz az egyes aloperátorok működése rendre:

DS: a tárfoglalás hosszának megállapítása (mindkét menetben); a tárgyprogrambejegyzés elkészítése (a második menetben),

DB,DW: a tárfoglalás hosszának kiszámítása (mindkét menetben); a tárfoglalás tartalmának és a tárgyprogrambejegyzésnek a meghatározása (a második menetben),

ORG: értékadás a beültetési számlálónak (LOC) (mindkét menetben); tárgyprogrambejegyzés készítése (a második menetben),

END: a forrásprogram befejeződését jelző indikátor (IEND) beállítása; tárgyprogrambejegyzés készítése (a második menetben),

EQU: ha az adott forrásnyelvi sor címkemezejében szereplő azonosítóhoz tartozó címketábla-bejegyzés szerint nem többszörös címkedefinícióról van szó, akkor az operandusmező értékével írja felül a címkeértéket a címketáblában, s beállítja a címkeflag végleges értékét (1 vagy 2),

SET: elvégzi az EQU direktívánál leírt műveleteket, továbbá ha a rendszer makrókifejtési állapotban van, megvizsgálja a modulcímketáblát is, van-e ugyanolyan nevű SET direktívával definiált azonosító, s ha ilyet talált, akkor az ehhez tartozó bejegyzésben is módosította a címkeértéket és címkeflaget,

IF: vagy az IF1 vagy az IF2 rendszerváltozó értékét növeli eggyel. IF2≠0 esetén a beskatulyázott IF direktívák operandusmezőit nem kell kiértékelni, IF2=0 esetén viszont minden beskatulyázott IF direktíva operandusa számít, függetlenül IF1 értékétől,

ENDIF: IF1 vagy az IF2 rendszerváltozó értékét csökkenti eggyel; az előbbit akkor, ha IF2 aktuális értéke nulla, különben IF2-t,

MACRO: ha a rendszer az A1 állapotban van: bejegyzést készít a makródefiníciós táblába, s a DEF rendszerváltozónak 1-es értéket ad,
ha a rendszer az A2 állapotban van: a DEF rendszerváltozónak 1-es értéket ad,
ha a rendszer az MK1i állapotok egyikében van: az operandusmező tartalmát (ha nem üres) átviszi a megszakítási hely és a makróparaméterek veremtárba és lezárja a verembejegyzést.
A második menetben a MACRO direktíva sorára nincs szükség, ezért egy különleges, „az operátor kész a kifejezést indító MACRO feldolgozása után" elnevezésű kimenő jel segítségével megakadályozzuk, hogy a sor felkerüljön a háttértárra.

MEND: ha a rendszer a DEF állapotban van: a DEF rendszerváltozónak 0 értéket ad (visszatérés az A1 vagy az A2 állapotba),
ha a rendszer az MK1i állapotban van: a kifejtési verembejegyzés alapján beállítja a visszatérési paramétereket és veremvisszaállítást hajt végre, azaz:

a) ha az előző, megszakított szinten lemezről történt a forrásprogram olvasása, akkor e megszakítási helyre vonatkozó lemezcímről visszaolvassa SPB1-re az előző szint következő sorát,

b) ha a visszatérés a nulladik szintre fog történni és a bemeneti periféria nem mágneslemez, akkor SPB3-ról visszatölti a következő sort SPB1-re,

c) lezárja az adott szinthez tartozó makrócímketáblát és kiviszi a lemezre, a makrócímketábla-verem megfelelő bejegyzése által meghatározott címre,

d) a megszakítási hely és makróparaméterek, valamint a makrócímketábla veremtárakban újból az előző szint bejegyzéseit érvényesíti,

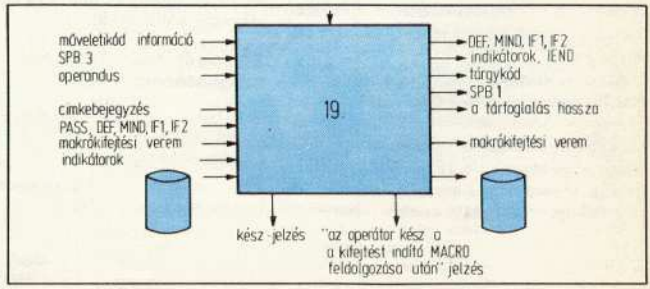
e) a MIND rendszerváltozó értékét eggyel csökkenti,

f) ha MIND új értéke nem 0, akkor a memóriacímke-tábla veremtár bejegyzése alapján visszahívja a DFB 4 pufferre a lemezről a korábban megszakított és most visszatért szinthez tartozó makrócímketáblát (illetve K>1 esetén ennek utolsó rekordját), ha a rendszer az MK2i állapotban van: az előző lépések közül csak d)-t, de csak a makrócímketábla veremtárra, e)-t és f)-et hajtja végre.

20. operátor: forrásprogramot mágneslemezre író *(5. ábra)*

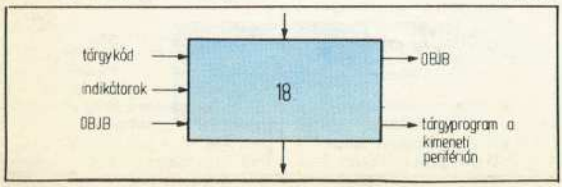
Feladata: az SPB2-n tárolt és az első menetben már feldolgozott forrásnyelvi sor háttértárra való felírását végzi el.

A sort az SPB2 területről átviszi a DFB2 pufferterületre, s megvizsgálja, hogy az betelt-e. Csak betelés esetén kezdeményezi ténylegesen a lemezre írási műveletet.

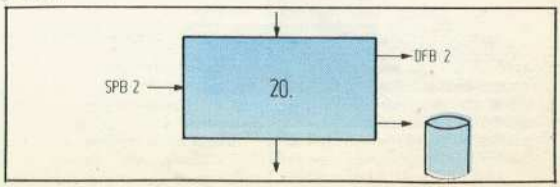


4. ábra

3. ábra



5. ábra



21. operátor: sorfeldolgozási ciklust lezáró *(6. ábra)*

Feladata: lezárja az adott SPB2-n tárolt sor feldolgozási folyamatát, és felkészíti a rendszert a következő, pillanatnyilag még az SPB1-en tárolt sor feldolgozására.

Valamennyi indikátort (köztük a hibaindikátorokat) alapállapotba állítja vissza, ha szükséges, akkor a tárfoglalás hosszának megfelelő értékkel növeli a beültetési számláló értékét, s a második menetben szóköz karakterekkel tölti fel a nyomtatópuffert (LPB).

22. operátor: a fordítás első menetét lezáró *(7. ábra)*

Feladata: az END direktíva beérkezése után lezárja a fordítás első menetét, és felkészíti a rendszert a második menet indítására.

Az operátor a következő műveleteket hajtja végre: a modulcímketábla utolsó (csonka) rekordját (MLB) lezárja, és kiviszi háttértárra — hacsak MLB nem üres; az összes indikátort alapállapotba állítja; a pufferterületek mutatóit, a lemez-munkaterületek címmutatóit a területek elejére állítja be; LMCT-t újra a makrócímketáblák számára fenntartott terület kezdőcímével tölti fel; a rendszerváltozóknak az induló A2 állapotnak megfelelő értékeket adja.

23. operátor: hibaüzenet-beállító *(8. ábra)*

Feladata: a forrásprogram címke-, műveletikód- és operandusmezőit feldolgozó operátorok által beállított hibaindikátorok alapján elhelyezi az esetleges hibaüzenetet a nyomtatópuffer (LPB) kijelölt területére.

Több hiba esetén az üzenet tartalmazhatja valamennyi hibát, vagy csak egy tetszőlegesen kiválasztottat. Magának az üzenetnek az összeállítása egy kódtábla segítségével történik, amely az egyes hibaindikátorokhoz tartozó szöveges vagy numerikus üzenetkódot tartalmazza.

24. operátor: fordítási listát készítő *(9. ábra)*

Feladata: a feldolgozott forrásprogramsort, a kísérő információkkal kiegészítve, kilistázza.

A fordítási lista tartalmát előre, a fordítóprogram tervezésekor kell rögzíteni. Egy lehetséges megoldásban soronként a következő mezők szerepelhetnek a listán:

— hibaüzenet (a 23. operátor tölti ki ezt a mezőt),
— sorszám: a forrásprogramban levő sorok sorszáma, makrókifejtéskor a sorszám nem növekszik,
— a beültetési számláló értéke: a tárgykód — ha van — első bájtjának fordítási kezdőcíme,
— a generált tárgykód — ha van,
— az operandus értéke — ha van,
— referenciamező: ha az utasításnak vagy direktívának volt operandusa és ebben egy címkére történt hivatkozás, akkor a hivatkozás helyét kapjuk meg ebben a mezőben,
— a forrásprogramsor tükörlistája (SPB2 tartalma).

Látható, hogy nem minden sornál lehet vagy kell valamennyi mezőt kitölteni, ennek szabályozását az operátor végzi el különféle, a korábban működött operátorok (16., 17., 19.) által beállított indikátorok alapján.

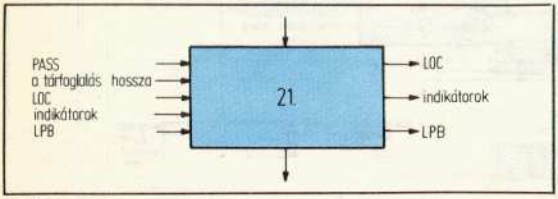
A *10.* és *11. ábrán* a most definiált 24 operátor segítségével összeállított kapcsolatábrát adjuk meg, amelyen a rendszer — a fordítóprogram — működése nyomon követhető. Mivel az egyes operátorokat jelző négyszögekbe csak a operátorok sorszámát jegyeztük be, az alábbi listában foglaljuk össze őket:

1. Paraméterlekérdező és kezdetiérték-beállító operátor
2. Forrásprogram-olvasó operátor nem lemezes bemeneti perifériáról
3. Forrásprogram-olvasó operátor mágneslemezről
4. Karakterlánc-mozgató operátor
5. Egy forrásprogramot mezőkre bontó operátor
6. Több-bites párhuzamos komparátor
7. Formális-aktuális paramétercserét végző operátor
8. Címketáblában kereső és címkeinformációt szolgáltató operátor
9. Címkeinformációt a címketáblában elhelyező operátor
10. Többszörös címkedefiniálást kezelő operátor
11. Címketáblát lemezre író operátor
12. Makróhívást felismerő operátor
13. Makrókifejtést indító operátor
14. Műveletikód-kereső és műveletkód-információkat szolgáltató operátor
15. Direktívafelismerő operátor
16. Operandusmező-kiértékelő operátor
17. Tárgykód-meghatározó operátor
18. Tárgyprogramkészítő operátor
19. Direktívafeldolgozó operátor
20. Forrásprogramot mágneslemezre író operátor
21. Sorfeldolgozási ciklust lezáró operátor
22. A fordítás első menetét lezáró operátor
23. Hibaüzenet-beállító operátor
24. Fordítási listát készítő operátor

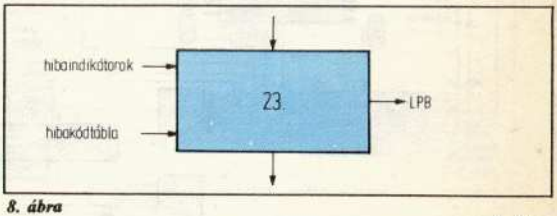
Az ábra áttekinthetősége érdekében egyes operátoroknál azokat a bemeneti és kimeneti változókat nem tüntettük fel, amelyeknek az adott menetben nincs szerepük.

Ahol egy operátor bemeneti vagy kimeneti változói között egy periféria szimbóluma jelenik meg, ott utalunk e periféria szerepére is. A szimbólumokba beírt betűk jelentése:

a: az eredeti vagy a kifejtett forrásprogram adatállománya mágneslemezen,
b: makródefiníció adatállománya mágneslemezen,

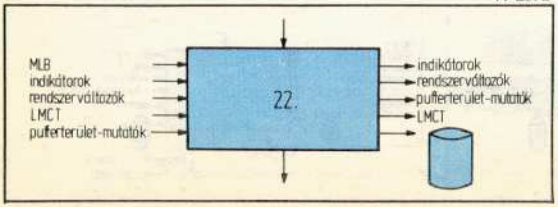


6. ábra

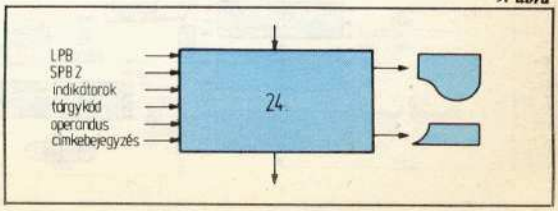


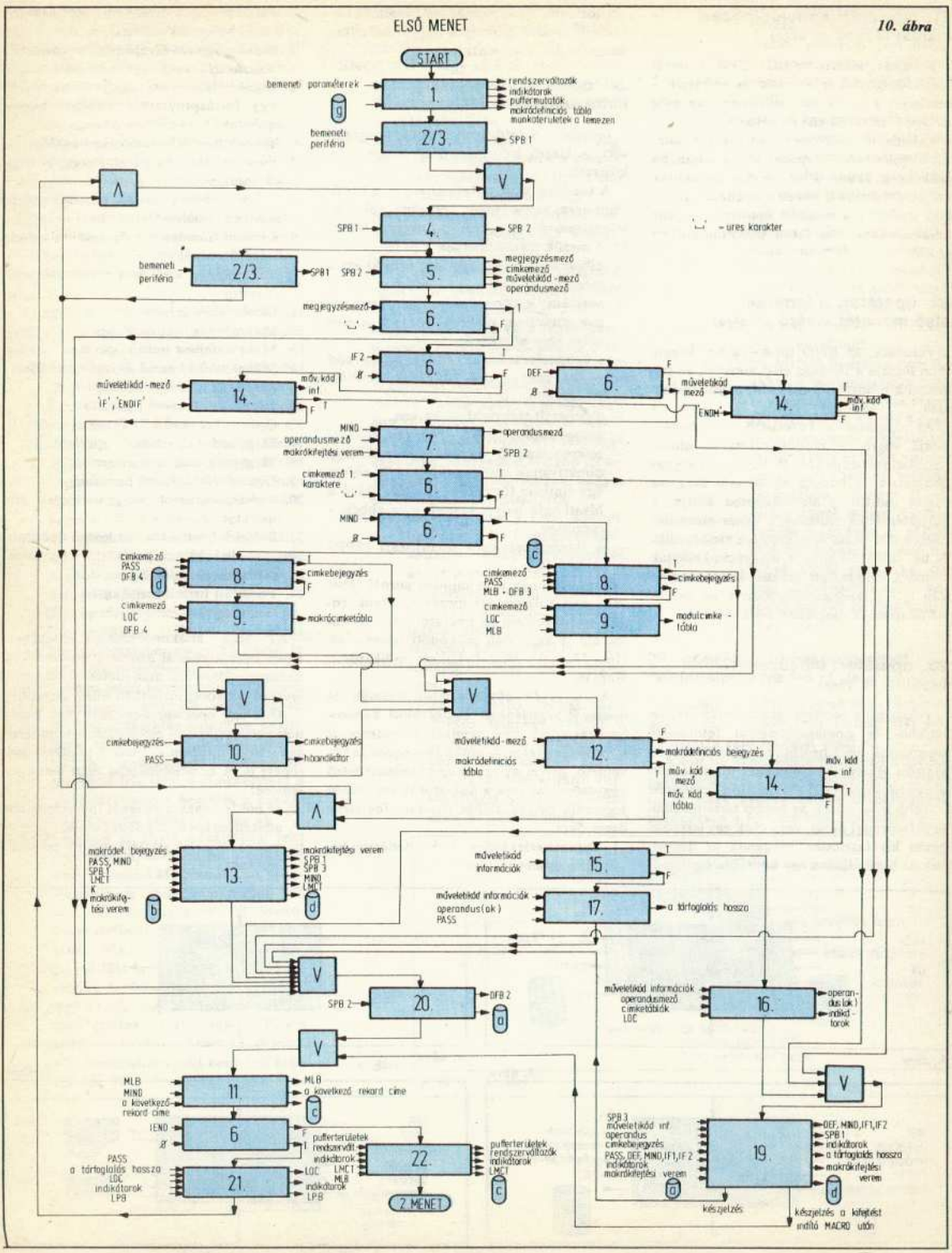
8. ábra

7. ábra



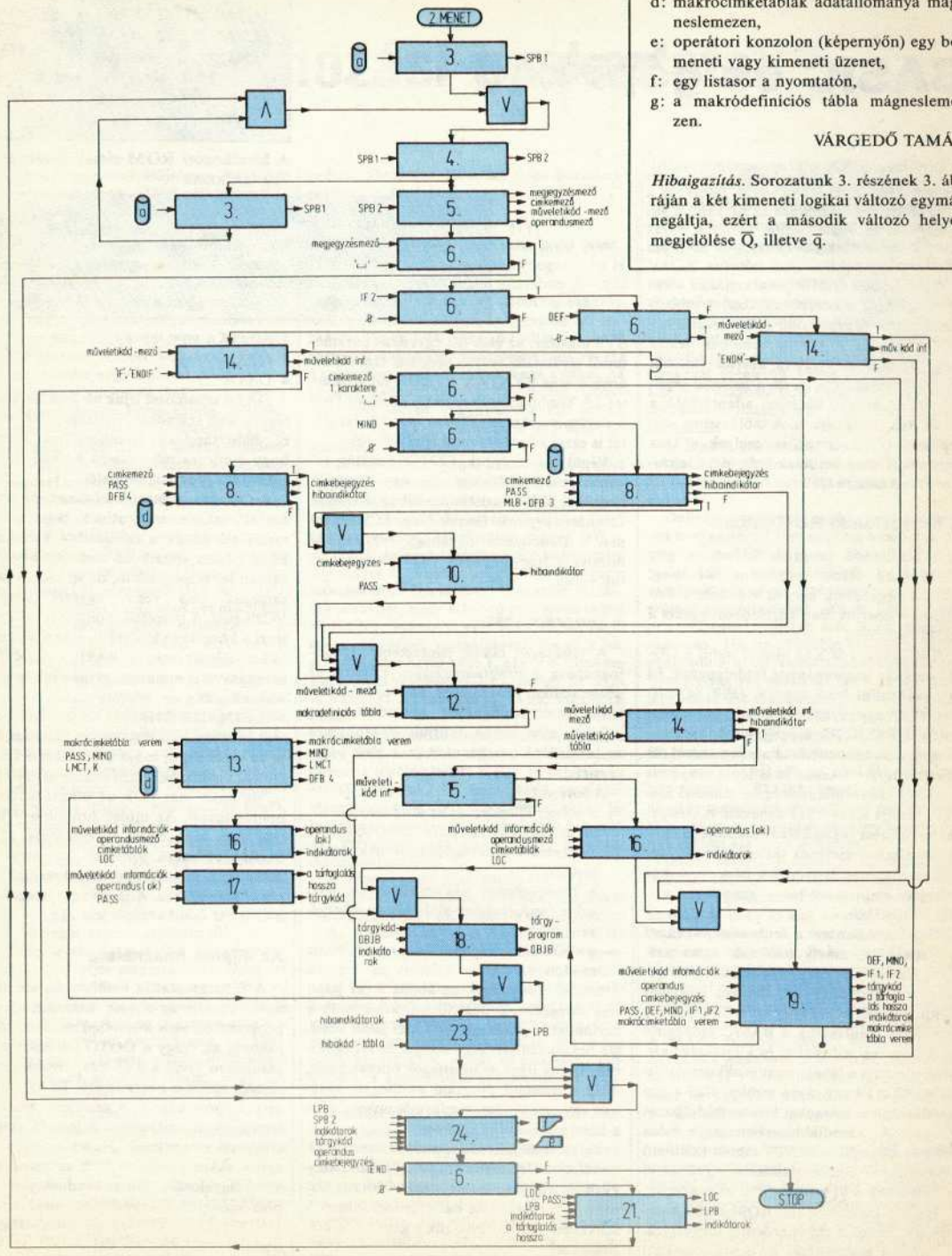
9. ábra




ELSŐ MENET
START
bemeneti paraméterek
rendszerváltozók
indikátorok
puffermutatók
makródefiniciós tábla
munkaterületek a lemezen
bemeneti periféria
SPB 1
SPB 2
megjegyzésmező
cimkemező
műveletikód-mező
operandusmező
IF 2
DEF
műveletikód mező
'IF', 'ENDIF'
mův. kód inf.
'ENDM'
MIND
operandusmező
makrókifejtési verem
cimkemező 1. karaktere
cimkemező PASS DFB 4
cimkebejegyzés
cimkemező LOC DFB 4
makrócimketábla
cimkemező PASS MLB + DFB 3
cimkemező LOC MLB
modulcimke-tábla
cimkebejegyzés
PASS
hibaindikátor
műveletikód-mező
makródefiniciós tábla
makródefiniciós bejegyzés
mův. kód mező
mův. kód tábla
mův. kód inf.
makródef. bejegyzés
PASS, MIND
SPB 1
LMCT
K
makrókifejtési verem
SPB 3
MIND
LMCT
műveletikód információk
műveletikód információk
operandus(ok)
PASS
a tárfoglalás hossza
SPB 2
DFB 2
műveletikód információk
operandusmező
cimketáblák
LOC
operandus(ok)
indikátorok
MLB
MIND
a következő rekord címe
IEND
a következő rekord címe
pufferterületek
rendszerváltozók
indikátorok
LMCT
MLB
PASS
a tárfoglalás hossza
LOC
indikátorok
LPB
2. MENET
SPB 3
műveletikód inf.
operandus
cimkebejegyzés
PASS, DEF, MIND, IF1, IF2
indikátorok
makrókifejtési verem
DEF, MIND, IF1, IF2
SPB 1
indikátorok
a tárfoglalás hossza
makrókifejtési verem
készjelzés
készjelzés a kifejtést indító MACRO után
'␣' = üres karakter

10. ábra

11. ábra



c: modulcímketábla adatállománya mágneslemezen,
d: makrócímketáblák adatállománya mágneslemezen,
e: operátori konzolon (képernyőn) egy bemeneti vagy kimeneti üzenet,
f: egy listasor a nyomtatón,
g: a makródefiníciós tábla mágneslemezen.

VÁRGEDŐ TAMÁS

Hibaigazítás. Sorozatunk 3. részének 3. ábráján a két kimeneti logikai változó egymás negáltja, ezért a második változó helyes megjelölése $\overline{Q}$, illetve $\overline{q}$.

BASIC és gépi kód

VC20	C64	C16
$C8A0	$A8A0	$8D4D
$C8A3	$A8A3	$8D50
$C96B	$A96B	$8E3E
$CD8A	$AD8A	$9714
$CEFD	$AEFD	$9491
$D7F7	$B7F7	$9DE4

A hivatkozott ROM-címek összehasonlító táblázata

```
$033C   JSR $CEFD     20 FD CE      828: 32 253 206
        JSR $CD8A     20 8A CD           32 138 205
        JSR $D7F7     20 F7 D7           32 247 215
        JMP $C8A3     4C A3 C8           76 163 200
                 ❶             ❷              ❸
```

Utoljára a 65XX mikroprocesszor család regisztereiről és az áprilisi számban megjelent segédlet használatáról volt szó, és megismerkedtünk néhány vezérlésátadó utasítással is. Most ezek gyakorlati alkalmazására látunk egy példát.

A kiszámított GOTO programjának elkészítését fogjuk a korábbinál részletesebben megismerni. A gépi kódú rutin BASIC betöltőprogramja a júniusi számban jelent meg. Itt csak a VC20-ra készült változat programozását írom le, de a *táblázat* segítségével az eljárás könnyen adaptálható a másik két géptípusra is. A táblázatban szerepelnek olyan címek is, melyek a kész programba nem kerülnek bele, de a leírásban hivatkozom rájuk.

A használandó ROM-rutinok

A készítendő program három, a gép ROM-jában tárolt szubrutint hív meg, majd a negyedikre ugorva visszatér a BASIC-be. Ismerjük meg közelebbről ezeket a ROM-rutinokat.

A VC20-on a $C8A0 címről indul a GOTO utasítás hagyományos feldolgozása. Itt egy szubrutint hívó utasítás (JSR $C96B) van, mely egy olyan alprogramot hív meg, amely a BASIC forrásprogramban található egész számot beolvassa a programból (és egyúttal természetesen át is lépi), majd ezt a számot címformátumú egész számmá alakítva tárolja a $14—$15 címeken. A címformátumú egész olyan kétbájtos, előjel nélküli egész szám, melynek tárbeli elhelyezésénél az alacsony helyértékű bájt — a már többször említett módon — megelőzi a magas helyértékűt.

Eljárásunkban ezt a szubrutint olyannal helyettesítjük, amely nemcsak egész számot, hanem tetszőleges aritmetikai kifejezést is ki tud értékelni. Ezt két lépésben hajthatjuk végre. Az elsőben a JSR $CD8A utasítással beolvassuk a BASIC szövegből az aritmetikai kifejezést, és kiértékelve azt, az eredményt a lebegőpontos akkumulátorba tesszük. Ez egy olyan terület, ahol a gép a valós típusú számokat tárolja feldolgozás közben. A második lépésben ezt a valós számot alakítjuk a $D7F7 címen található szubrutinnal címformátumúra, egyúttal el is helyezzük a $14—$15 című bájtokon.

Ezeken kívül még egy ROM-rutint használunk. Ennek a címét érdemes megjegyezni, mert a paraméteres SYS utasításoknál gyakran találkozunk vele: $CEFD. Ez a SYS utasítást az első (itt egyetlen) paramétertől elválasztó vessző meglétét figyeli, hiánya esetén SYNTAX ERROR hibaüzenetet ad. Többparaméteres SYS utasításoknál a paramétereket elválasztó vesszők meglétét is ezzel a szubrutinnal ellenőrizzük.

Végül visszatérünk GOTO utasítás további feldolgozásához. Ez úgy történik, hogy egy JMP utasítással a kihagyott utáni utasításra ugrunk. Ennek címe: $C8A3. Innen a feldolgozás ugyanúgy folytatódik, mintha a hagyományos GOTO-t használtuk volna.

A programozás

A szükséges címek ismeretében hozzáfoghatunk a programozáshoz. Először az *ábrán* látható programnak az 1-es számmal jelölt részét készítjük el. Az indulási cím felírása után sorba felírjuk a szükséges szubrutinhívó utasításokat és a záró ugróutasítást.

A sorrend:

— a vesszők ellenőrzése és átlépése,
— az aritmetikai kifejezés kiértékelése és elhelyezése a lebegőpontos akkumulátorban,
— a lebegőpontos akkumulátorban lévő szám címformátúra alakítása és elhelyezése,
— a GOTO feldolgozása a hagyományos módon.

Ezután következik az ábrán 2-vel jelölt rész elvégzése. A segédletből kikeressük a hárombetűs emlékeztetőkóddal jelölt utasítás hexadecimális gépi kódját, és feljegyezzük. Utána írjuk az utasítások operandusát, mely a mostani program mindegyik sorában egy-egy tárcím, címformátumban, azaz a bájtok sorrendjét felcserélve.

Befejezésül a hexadecimális számokat kicseréljük a decimális megfelelőjükkel, ahogyan az ábra harmadik részén látható. Ezután egy rögtönzött betöltőprogrammal a szükséges helyre bevisszük a gépi kódú utasításokat:

```
1 FOR I=828 TO 828+11
2 READ A : POKE I,A
3 NEXT
4 DATA . . .
```

A DATA utasításba írjuk az ábra harmadik részén lévő számokat. Futtatás előtt célszerű háttértárolóra kimenteni az egészet, hogy esetleges hiba esetén ne kelljen újra kezdeni a program bevitelét.

A C64-en ugyanezt a módszert alkalmazhatjuk, csak a szubrutinok címe más, a megfelelő címek a táblázatból kereshetők ki. A C16-ra vonatkozó címeket is a táblázatban lehet megtalálni, de itt csak a programozás első része egyezik meg a VC20-éval. A beépített monitor feleslegessé teszi a következő lépéseket, a kézi asszemblálást. Hívjuk meg a BASIC MONITOR utasításával a monitort, majd írjuk be a következő sort:

```
A 033C JSR $9491
```

A monitor lefordítja ezt az utasítást, kiírja az eredményt, majd a következő szabad címet. Vigyük be a következő utasításokat is, de most már csak az utasításkódot az operandussal. Az utolsó programsor bevitele után kiírt címre válaszoljunk a RETURN billentyű azonnali lenyomásával, aztán az X monitorparanccsal térjünk vissza a BASIC-hez. A program kimentéséről a C16-nál majd később lesz szó.

Az eljárás használata

A futtatás után a betöltőprogram törölhető. Ezután az eljárás használható akár programból, akár közvetlen módban. A különbség az, hogy a GOTO sor-szám alakú utasításon kívül a SYS 828, aritmetikai kifejezés alakú is használható. Ez az utasítás arra a sorra adja át a vezérlést, amelynek sorszáma az aritmetikai kifejezés aktuális értékével megegyezik. Ha ez az érték nem egész, akkor az eljárás csak az egész részét veszi figyelembe. Ha az eredményként kapott számnak megfelelő sor nem létezik, UNDEF'D STATEMENT hibajelzést kapunk.

BARNA LÁSZLÓ

Z80 programozási gyakorlatok

A gépben számokat előjelesen és előjel nélkül ábrázolhatunk. Előjel nélkül rendkívül egyszerűen, a szokásos kettes számrendszerbeli alaknak megfelelően; előjelesen ún. kettes komplemens ábrázolási móddal. Ha n biten kettes komplemens ábrázolással akarunk felírni m számot, akkor ha az nem negatív, a kettes komplemens alak megegyezik az előjel nélküli ábrázolásmóddal; ha m negatív, akkor az $m+2^n$ szám kettes számrendszerbeli alakjával. Ennek lényegét mutatja négy bitre az *1. ábra.*

Természetesen lehet öt, hat ... stb. biten is ábrázolni kettes komplemens kódban számokat. Célszerű a központi egység (CPU) belső felépítése és utasításkészlete miatt 8 vagy 16 biten ábrázolni az egészeket, és mivel nyolc bit csupán a $(-128,127)$ intervallumba eső számokat engedi meg, ezért a legtöbb interpreter és fordító az egész számokat 16 biten ábrázolja.

Sorozatunk e cikkében azt fogjuk megnézni, hogy hogyan lehet az ilyen alakban felírt számokkal számolni. Ha az ADD vagy SUB utasítást használjuk, és nincs túlcsordulás, azaz az eredmény a $(-2^{15},2^{15}-1)$ intervallumba esik, $n=8$ vagy $n=16$, akkor helyes eredményt kapunk az operandusok előjelétől függetlenül. Egy szám előjelét megállapíthatjuk csak a legfelső bit vizsgálatával, ugyanis ha ez egy, akkor a szám negatív, ha nulla, akkor a szám pozitív vagy nulla.

Érdekes azt is megfigyelnünk, hogy ha egy szám minden bitjét az ellenkezőjére váltjuk, majd a számhoz hozzáadunk egyet, akkor eredményül pontosan a szám mínusz egyszeresét kapjuk *(2. ábra)*, egy esetet kivéve. Ez az eset a $-2^{15}(n-1)$ (n a bitek száma), ugyanis a pozitív számokat csak $2^{(n-1)}-1$-ig tudjuk ábrázolni. Ha $-2^{(n-1)}$ számra végezzük el a műveletet, akkor önmagát kapjuk eredményül.

Néhány gyakorlat a kettes komplemens kódábrázolás teljes megértéséhez. Írjuk fel négy- és nyolcbites alakban -1-et és -8-at. Számoljuk ki a megadott eljárással az ellentettjüket. Annak, aki most találkozott először ezzel az ábrázolással, azt ajánlom, hogy próbálja meg előbbi állításaimat bizonyítani, és gondolja végig, hogy IX vagy IY regiszterrel való címzésnél hogyan adódik ki a végleges cím, illetve hogy a relatív ugrásoknál hogyan kell kiszámítani az ugrás hosszát. Most pedig nézzük meg a példákat.

1. Írjunk szubrutint, amely HL-ben adott előzetes számnak kiszámítja az abszolút értékét! Carry jelezze a hibát!

A HL regiszterpárban, 16 biten $(-2^{15}, 2^{15}-1)$ intervallumba eső egész számokat tudunk ábrázolni. Ez a két szám hexadecimálisan és binárisan

$+2^{15}-1=7FFFh=$ /
0111 1111 1111 1111
-2^{15} $=8000h=$ 1000 0000 0000 0000

A HL-ben levő szám akkor negatív, ha H hetes bitje 1. A biteket általában jobbról balra nullától hétig vagy tizenötig számozzuk, így a nullás bit 2^0, az egyes bit 2^1 stb. helyiértéknek felel meg. Az abszolút érték fogalmát definiáljuk a szokásos módon. Ha a szám nem negatív, akkor maga a szám, ha negatív, akkor a mínusz egyszerese. Gondban csak akkor vagyunk, ha HL értéke $-2^{(n-1)}$. Ebben az esetben kell hibát jeleznünk. *(1. program)*

Ha HL nem negatív, akkor a carry bitet töröljük és visszatérés. Ha negatív, akkor átváltjuk az ellentettjére. Ezt két lépésben tehetjük meg, külön az alsó, majd a felső bájtot negálva, és növelve eggyel HL-t. A -2^{15} számot kivéve az eredmény mindig helyes és pozitív. -2^{15}-re a szubrutin lefutása után HL értéke változatlan marad, így tehát azt, hogy HL-ben ez a szám volt-e eredetileg, könnyen eldönthetjük az eredmény előjelének vizsgálatával. Ha ez a bit, a H regiszter hetes bitje 1, akkor a szám negatív, pontosabban -2^{15}, és így az SLA A hatására a carry bit egy lesz, különben nulla.

Az előző részben arról is volt szó, hogy a carry bitet az OR A vagy az AND A utasítással is törölni lehet. Vajon itt miért nem ezt alkalmaztuk? Ha felcseréljük az előjelvizsgálatot és a carry bit törlését, akkor lehet-e ezen utasítások valamelyikét használni? Próbáljuk meg elkészíteni a program dokumentációját!

2. Írjunk szubrutint, amely HL és DE regiszterpárokban adott előjeles számokat összeadja! Carry bit jelezze a túlcsordulást!

Az összeadásra lehet használni az ADD utasítást, csupán a túlcsordulást kell figyelni. Ezt az előjelek vizsgálatával tehetjük meg. Ha a két szám előjele különböző, akkor nem lehet túlcsordulás, mert az eredmény nagyobb mint a negatív, és kisebb mint a nem negatív operandus. Ha a két előjel megegyezik, akkor az eredmény előjelének is ugyanannak kell lennie. *(2. program)*

A kizáró VAGY-ok alkalmazásakor koncentráljunk csak a hetes, azaz előjelbitre. Ha a két előjel egyezik, vagyis H és D regiszterek előjelbitje azonos, akkor a kizáró vagy művelet hatására az akkumulátor hetes bitje nulla, különben egy. Ez a bit kerül a carry bitbe. Ha különböző a két előjel, akkor összeadjuk a két számot, és mivel bizonyos, hogy nem volt hiba, majd a visszatérés előtt töröljük a carry bitet. Ha egyezik a két előjel, akkor az LB címkénél folytatódik a futás. Itt az akkumulátor hetes bitjébe kerül a közös előjel, majd az összeadás után, az eredmény előjelével vett kizáró VAGY művelet hatására az akkumulátor hetes bitje egy lesz, ha a két előjel különböző, nulla, ha nem változott az előjel, azaz ha nincs túlcsordulás, és a shiftelő utasítás ezt a bitet viszi a carry bitbe. A rutin megváltoztatja az A regiszter tartalmát. Az LB címkével megjelölt sorba írhattunk volna LD A,D utasítást is, a későbbiek folyamán azonban felhasználjuk, hogy nem ezt választottuk.

3. Írjunk szubrutint, amely HL-ből kivonja DE-t! Tekintsük HL-t és DE-t előjeles számoknak! Carry bit jelezze a túlcsordulást!

Definiáljuk úgy a kivonást, mint az ellentett hozzáadását! Egy esetben nem tehetjük meg ezt, ha DE értéke $-2^{(n-1)}$. Ezt az esetet külön kell kezelnünk. Szükségünk van tehát egy olyan szubrutinra, amely egy számot az ellentettjére vált, és jelzi, ha éppen $-2^{(n-1)}$-ről van szó. Ilyen rutinunk már van. Ha az abszolút értéket számító programba a MINUSZ címkével jelölt helyen lépünk be, akkor az HL-t az ellentettjére váltja. Ha ezután carry=0, akkor használjuk a PLUS rutint. Ha carry=1, akkor is össze kell adni HL-t és DE-t, de DE előjelét pozitívnak kell tekintenünk annak ellenére, hogy D előjelbitje egy. Ha az összeadó rutinba a PLUS1 címnél lépünk be, és az akkumulátorba H regiszter értékét töltjük, akkor a szubrutin mindenütt pozitívnak tekinti DE-t. *(3. program)*

Mivel MINUSZ rutin HL-t változtatja, nekünk viszont DE mínusz egyszeresére van szükségünk, ezért a szubrutin meghívása előtt és után ki kell cserélni HL és DE regiszterpárokat. A csere nem változtatja meg carry bit értékét, és ha éppen -2^{15} a kivonandó, akkor az akkumulátor hetes bitjébe H előjelét tesszük, ami ha nulla, akkor egyezik, ha egy, akkor különbözik DE előjelétől, mivel DE-t pozitívnak tekintjük. A PLUS1 belépési címnél ennek a bitnek ugyanez a jelentése, amikor az összeadó rutin fut. A JP PLUS1 utasítás ugyanazt teszi, mint az egymást követő CALL PLUS1 , RET utasítások. A program megváltoztatja az A regiszter tartalmát.

4. Írjunk szubrutint, amely HL értékét megszorozza tízzel. Carry jelezze a túlcsordulást! Tekintsük HL-t előjel nélkülinek!

A legegyszerűbb megoldás, ha HL értékét áttöltjük BC-be vagy DE-be, és kilencszer hozzáadjuk HL-hez. Ez a megoldás azonban lassú. Ha HL-hez önmagát adjuk,

azzal kettővel szoroztuk. Ha ismét hozzáadjuk önmagához, akkor az eredeti érték négyszeresét, ha ismét hozzáadjuk önmagához, akkor nyolcszorosát stb. kapjuk. Ilyen módszerrel ki tudjuk számítani HL bármely kettőhatványszorosát. A tíz nem kettőhatvány, de mint bármely szám, felírható kettőhatványok összegeként, azaz felírható kettes számrendszerben. A $10=2^3+2^1$ összefüggést felhasználva gyorssá válik a tízzel való szorzás. HL-t szorozzuk kettővel, azaz hozzáadjuk önmagához, ezt az értéket eltesszük BC-be, majd még kétszer szorozzuk kettővel, és hozzáadjuk a BC-be eltett értéket. Ha bárhol túlcsordulás van, akkor HL tízszerese nem fér el 16 biten. Bármelyik RET utasításnál is ér véget a program, a carry bit jelzi a túlcsordulást. *(4. program)*

A szubrutin felhasználja BC regiszterpárt is.

A most következő program előtt meg kell hogy álljak egy pillanatra. Az első részben gépfüggetlen példákat igértem. A továbbiakban azonban szükségessé válik valamilyen gépi reprezentációhoz való kötődés. Így a következő példában feltételezzük egy PRNA nevű szubrutin létezését, amely a regiszterek értékének megváltoztatása nélkül az akkumulátorban lévő ASCII karaktert kiírja például a képernyőre. Ez a szubrutin gépfüggő. A gép leírásában többnyire megtalálható valamilyen formában, hogy hogyan kell egy ilyen rutint megírni.

5. Írjunk szubrutint, amely HL-t mint előjel nélküli egész számot kiírja a standard kimenetre decimálisan!

A szám maximum $2^{15}-1$ lehet. Ez decimálisan 65 535, tehát kisebb, mint százezer. Az algoritmus az első közelítésben a következő: addig vonunk le HL-ből 10000-et, hogy az még éppen negatívvá ne váljon. Ahányszor ezt megtettük, az a tízezres helyiértéken álló számjegy. Ezt kiírjuk. HL-ben ezután az eredeti érték 10000-rel való osztási maradéka van. Ezt a műveletet megismételjük ezer, száz, tíz levonogatásával is, majd a maradékot, ami az egyesek helyén álló számjegy, is kiíratjuk. Mit ad eredményül ez az algoritmus, ha HL-ben 32 van? A standard kimeneten 00032 fog megjelenni.

Valahogy el kellene kerülni, hogy a bevezető nullák kiíródjanak. Az nem lehet megoldás, hogy ha a kurrens digit nulla, akkor nem íratjuk ki, mert így 102 helyett 12 jelenne meg. Akkor nem kell csak kiíratni a nulla karaktert, ha az bevezető nulla, azaz még nem írtunk ki egy számjegyet sem, de ettől függetlenül az utolsó számjegyet mindenképpen ki kell íratnunk. Így HL=0 esetén is helyesen fog működni a program. A C regiszterben tároljuk, hogy volt-e már számjegykiírás. A program elején C-t nullázzuk, ez jelzi, hogy még nem írattunk ki egy karaktert sem. Minden számjegy kiírása után C-t növeljük eggyel, így a szubrutin lefutása után C a kiírt jegyek számát mutatja. Mivel ez maximum öt lehet, így nem kell attól félni, hogy a növelés következtében C nullává válik. Az akkumulátort azért nullázzuk minden kiíratás előtt, mert ebben a regiszterben számoljuk, hogy DE-t hányszor tudjuk levonni HL-ből. *(5. program)*

A számláláshoz felhasználtuk a már megírt összehasonlító rutint. Ennek visszatértekor carry=0, ha HL nagyobb vagy egyenlő mint DE; ekkor le kell vonni HL-ből DE-t. Mivel carry=0, ezért nem kell törölni. Ha carry=1, azaz nem lehet többször levonni, akkor meg kell vizsgálnunk, hogy nulla bájtról van-e szó. Ha nem, akkor A-t átalakítjuk ASCII kódra, kiíratjuk a karaktert, és növeljük C-t. C-ben mindig a már kiírt karakterek száma van. Ha nulla bájtról van szó, akkor a NULL címkénél folytatódik a program. Ha még nem írtunk ki egy jegyet sem, akkor most a nullát sem írjuk ki, ha már volt, akkor kiírjuk. Az utolsó jegy mindenképpen kiíródik, mert itt nincs szükség számlálásra, és így a vizsgálatot is kihagyhatjuk. Tízezret, ezret, százat a DE regiszterpárba töltöttünk, tizet elég az E-be, mert ekkor D-ben már nulla van. Felhasználtuk azt is, hogy a feltételezett PRNA rutin nem változtatja meg a regiszterek értékeit. A program az A,C,D,E,H és L regiszteret használja fel. Meg lehet írni valamivel rövidebben is a programot, az összehasonlító rutin hívása nélkül is, így valamivel gyorsabb.

6. Írjunk szubrutint, amely HL-t mint előjeles egész számot kiíratja a standard kimenetre decimálisan!

Az előző program megírása után ezt a problémát már nem nehéz megoldani. Ki kell íratni HL előjelbitjének megfelelően egy "+" vagy "−" jelet, esetleg csak szóközt, meghívni az abszolút értéket számító szubrutint, nem kell törődni a túlcsordulással sem, és kiíratni HL-t mint előjel nélküli egész számot. *(6. program)*

A program A,C,D,E,H és L regisztereket használja fel.

Assembly programozás során, de főleg gépi kódu programozásnál gyakran találkozunk a tizenhatos vagy más néven hexadecimális számrendszerrel. A kódtáblázatok ilyen alakban adják meg a kódokat, és a legtöbb monitorprogram is ezt a számrendszert használja. A következő két feladat ehhez a számrendszerhez kapcsolódik.

7. Írjunk szubrutint, amely a HL címen lévő kétbájtos számot kiírja a standard kimenetre hexadecimális alakban.

1 Program

```
ABS     BIT 7,H ;az előjel vizsgálata
        SCF     ;carry
        CCF     ;törlése
        RET Z   ;ha pozitív volt a szám
MINUSZ  LD  A,L
        CPL     ;alsó bájt
        LD  L,A ;negálása
        LD  A,H
        CPL     ;felső bájt
        LD  H,A ;negálása
        INC HL
        LD  A,H ;a legfelső bitet
        SLA A   ;carrybe toljuk
        RET
```

2 Program

```
PLUS    LD  A,D     ;az előjelek
        XOR H       ;vizsgálata
PLUS1   SLA A
        JR  NC,LB   ;egyeznek az előjelek
        ADD HL,DE
        AND A       ;carry törlése
        RET
LB      LD  A,H     ;a közös előjel
        ADD HL,DE
        XOR H       ;az eredmény előjele
        SLA A       ;carry flagbe
        RET
```

3 Program

```
KIV     EX   DE,HL      ;ellentettre
        CALL MINUSZ     ;váltás
        EX   DE,HL
        JP   NC,PLUS    ;sima összeadás
        LD   A,H        ;előjel bit
        JP   PLUS1
```

4 Program

```
TEN     ADD HL,HL ;szorzás kettővel
        RET C     ;carry=1 =>túlcsordulás
        LD  B,H   ;BC-be az érték
        LD  C,L   ;kétszerese
        ADD HL,HL ;négyszeres
        RET C
        ADD HL,HL ;nyolcszoros
        RET C
        ADD HL,BC ;tízszeres
        RET
```

5 Program

```
OUTNUM XOR  A       ;C és A
       LD   C,A     ;törlése
       LD   DE,10000
       CALL OUTDIG  ;tízezresek kiíratása
       XOR  A
       LD   DE,1000
       CALL OUTDIG  ;ezresek kiíratása
       XOR  A
       LD   DE,100
       CALL OUTDIG  ;százasok kiíratása
       XOR  A
       LD   E,10    ;D=0
       CALL OUTDIG  ;tízesek kiíratása
       LD   A,L
       JR   LAST    ;egyesek kiíratása
SZAML  SBC  HL,DE   ;carry=0
       INC  A       ;számlálás
OUTDIG CALL COMP    ;hasonlítás
       JR   NC,SZAML
       OR   A       ;nulla számjegy ?
       JR   Z,NULL  ;igen
LAST   ADD  A,'0'   ;konverzió ASCII-re
OUT    CALL PRNA    ;kiíratás
       INC  C       ;számjegyek számlálása
       RET          ;következő
NULL   LD   A,C     ;ki kell-e
       OR   A       ;írni?
       RET  Z       ;zero=1, nem
       XOR  A       ;A-ba újra nulla
       JR   LAST
```

```
1000  =  -8
1001  =  -7
1010  =  -6
1011  =  -5
1100  =  -4
1101  =  -3
1110  =  -2
1111  =  -1
0000  =  +0
0001  =  +1
0010  =  +2
0011  =  +3
0100  =  +4
0101  =  +5
0110  =  +6
0111  =  +7
```

1. ábra.

```
   01100111
+  10011000
-----------
   11111111

A+Ā = 2ⁿ-1
Ā+1 = 2ⁿ-A
Ā+1 = -A
```

2. ábra

```
 6 Program
----------

OUTDEC BIT  7,H     ;előjelvizsgálat
       JR   Z,POZ
       LD   A,'-'   ;mínusz előjel
IR     CALL PRNA    ;előjel kiírás
       CALL ABS     ;abszolútérték
       JP   OUTNUM  ;szám kiírása
POZ    LD A,'+'     ;lehet szóköz is
       JR IR
```

```
 7 Program
----------

OUTHEX INC  HL
       LD   B,2     ;két bájt
BK     RLD          ;akkumulátorba
       CALL HEX     ;felső fél bájt
       RLD          ;akkumulátorba
       CALL HEX     ;alsó fél bájt
       DEC  HL      ;alsó bájtra visszalépés
       DJNZ BK
       INC  HL
       RET
HEX    LD   C,A     ;A elmentése
       AND  0FH     ;maszkolás
       CP   10
       JR   C,SKIP  ;ha kisebb mint 10
       ADD  A,'A'-'0'-10
SKIP   ADD  A,'0'
       CALL PRNA    ;kiírás
       LD   A,C     ;eredeti tartalom
       RET
```

```
 8 Program
----------

OUTHLX PUSH HL     ;a stack tetejére
       LD   HL,0   ;HL-be
       ADD  HL,SP  ;SP értéke
       CALL OUTHEX
       POP  HL     ;HL vissza
       RET
```

A feladat megoldásához először próbáljunk meg egy segédprogramot írni, amelynek bemenő adata egy "0" és "F" közé eső egész szám az akkumulátorban, és amely ezt a számot kiírja hexadecimálisan. Ehhez át kell alakítani a számot a megfelelő ASCII kódra. Ha a szám kisebb mint tíz, akkor "0"-t kell hozzáadni, ha nagyobb mint kilenc, akkor „A" – 10-et. Ezt csinálja a program HEX címkétől kezdődő része. Ezek után nincs más dolgunk, mint az akkumulátor alsó négy bitjébe belevarázsolni egymás után először a felső, majd az alsó bájt rendre felső, majd alsó félbájtját, és mindig meghívni a HEX címkétől kezdődő részt. Eltolási műveletre legalkalmasabb az RLD utasítás. Általában, ha azt külön ki nem hangsúlyozzák, akkor egy Z80-as rendszerben mindig az alacsonyabb címen van az alacsonyabb, a magasabb címen a magasabb helyiértékű bájt. Ez csupán szokás, de ezt segítik a központi egység 16 bites adatmozgató utasításai is, érdemes ehhez a szokáshoz ragaszkodni. Ha a HL címen levő számról beszélünk, akkor HL az alacsonyabbik értékű bájtra mutat, ezért kell HL-t növelni a szubrutin elején. Az AND műveletre azért van szükség, mert nem tudjuk, hogy mi volt az akkumulátor eredeti tartalma, és nekünk az kell, hogy a felső négy bit nulla legyen. Az AND n utasítás eredménye, ha n valamely kettőhatvány mínusz egy, mindig az akkumulátor eredeti tartalma n+1-gyel való osztási maradéka. *(7. program)*

A program használja A,C,H és L regisztereket, de csak C értékét írja felül. Próbáljuk meg ezzel a szubrutinnal kilistáztatni a ROM egy darabját! Miért nem lehet?

8. Írjunk szubrutint, amely a HL regiszterpárban lévő számot kiírja a standard kimenetre hexadecimális alakban! Használjuk fel az előző szubrutint!

Ha fel akarjuk használni az előző szubrutint, akkor ahhoz HL értékét le kell tenni valamilyen memóriahelyre, és a címet tölteni HL-be. Legyen ez a memóriahely, ahova HL-t tesszük, a verem (stack). A címe benne van a veremmutatóban. Valahogyan át kell juttatni HL-be. Mivel erre nincs utasítás, ezért két lépésben valósítjuk meg. *(8. program)*

Érdemes a program kapcsán megjegyezni, hogy a veremmutató mindig az utoljára eltett bájtra mutat. HL-t nem csupán azért kell kivenni a veremből, hogy megmaradjon az eredeti értéke, hanem azért is, hogy a RET utasítás a megfelelő értéket találja a veremben.

VERHÁS PÉTER

A CAD alapelemei Commodore mikroszámítógépeken

A háromdimenziós ábrázolás

A CAD (Computer Aided Design) rendszerek fogalmán tágabb értelmezésben a számítógéppel segített tervezést értjük, melybe a rajzi megjelenítés, a térbeli ábrázolás, a formatervezés éppúgy beletartozik, mint a statikai számítások, költségvetés készítése vagy az anyagmennyiségek meghatározása. Szűkebb értelmezésében a CAD fogalmán a rajzi megjelenítést értjük: 2 dimenziós alaprajzok, nézetrajzok, metszetek (2D-rendszerek), formatervek, 3 dimenziós tömegvázlatok (3D-rendszerek) számítógépes előállítását.

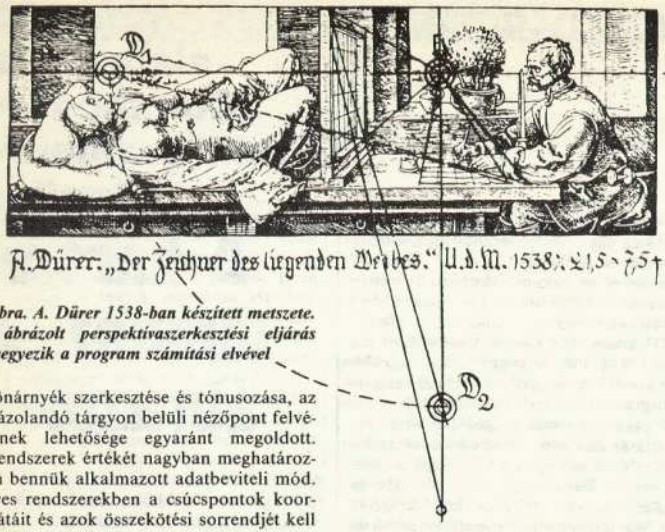
Annak ellenére, hogy a CAD rendszerek többsége nagy központi egységgel, 16 vagy 32 bites processzorral, nagy kapacitású gyors háttértárolókkal, nagy felbontású grafikával (1024*1024 pont) rendelkező számítógépekre készült, több 8 bites mikrogépre írt, sikeres CAD rendszer is bizonyítja, hogy mikroszámítógépekre is lehet professzionális színvonalú programokat készíteni (SCRIBE—Apple II/e).

A háromdimenziós tömegek és terek kétdimenziós síkon való ábrázolásának legszemléletesebb, a műszaki rajzok és tervek világában nem járatos felhasználók számára is közérthető módja az axonometria és a térbeli képi látvánnyal megegyező perspektíva.

Évszázados út volt, míg a perspektív ábrázolás törvényeit felismerték, és az összefüggések leírhatóvá váltak. A XV. században jelentek meg az ezzel kapcsolatos első munkák, melyeket a mai napig igen sok további tanulmány, könyv követett.

A pontos perspektív rajzok előállításának eszköze a szerkesztés volt, bár történtek kísérletek mechanikus szerkesztőgépek kifejlesztésére is. Annak ellenére, hogy a perspektív ábrázolás matematikai összefüggéseit már régen megfogalmazták, a számítás, mint perspektivaszerkesztési eljárás csak a számítógépek megjelenésével kaphatott létjogosultságot, általános gyakorlati felhasználása pedig a grafikus printerek és a plotterek elterjedésével vált lehetővé.

A 3 dimenziós programrendszerek bonyolultsági foka különböző. Az igényesebb rendszerekben a takart élek törlése, vetett



1. ábra. A. Dürer 1538-ban készített metszete. Az ábrázolt perspektívaszerkesztési eljárás megegyezik a program számítási elvével

és önárnyék szerkesztése és tónusozása, az ábrázolandó tárgyon belüli nézőpont felvételének lehetősége egyaránt megoldott. A rendszerek értékét nagyban meghatározza a bennük alkalmazott adatbeviteli mód. Egyes rendszerekben a csúcspontok koordinátáit és azok összekötési sorrendjét kell megadni, máshol az ábrázolandó testet éleivel, határoló lapjaival vagy egyszerű tömegek összegezésével definiálhatjuk.

A *programlista* egy egyszerű 3 dimenziós program szervezését és alapvető rutinjait tartalmazza. A program megírása döntő részben a Commodore 64 BASIC 2.0 verziójával történt, de e nyelv fogyatékosságai miatt a közismert Simon's BASIC bővítést is felhasználtuk.

A program modulokból áll, melyek módosíthatók, cserélhetők. A korlátozott terjedelem miatt az egyszerűbb rutinokat nem közöljük, megírásuk azonban nem nehéz feladat. A program alkotóegységei a következők.

1. Menü

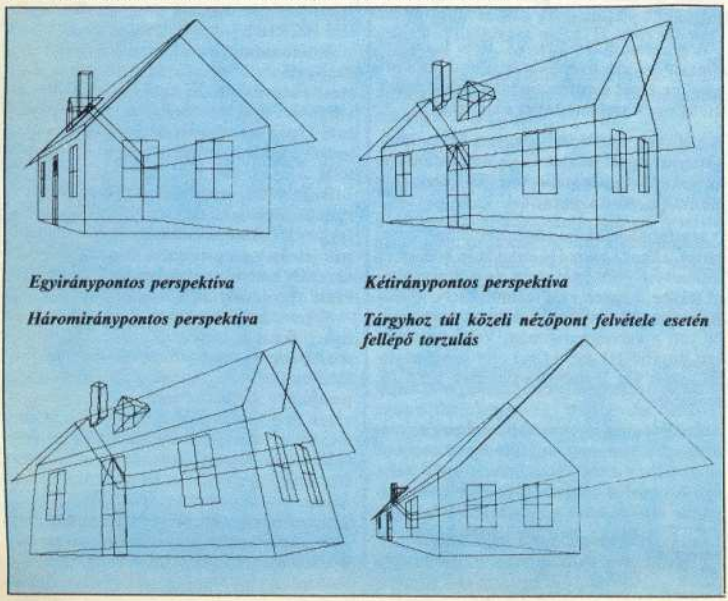
A programot szervező egység. Az aktuális modul inverz alakban jelenik meg, ennek változtatása a kurzormozgató, hívása a RETURN billentyűvel történik.

2. Alapadatok beírása

A néző és a főpont koordinátáinak megadását szolgálja. Ezen adatok és az ábrázolandó pontok megadása ugyanabban a tetszőleges koordinátarendszerben történik.

A néző koordinátáinak megadása:

2. ábra. Lakóépület különböző nézőpontokból felvett perspektív rajzai




```
10 REM KURZORVEZERLO JELEK
20 REM CLR              S
30 REM HOME             S
40 REM KURZOR JOBBRA    J
50 REM KURZOR BALRA     B
60 REM KURZOR LE        L
70 REM KURZOR FEL       F
80 REM INVERZ BE        S
90 REM INVERZ KI        C
100 PH=30:DIM X(PH),Y(PH),Z(PH)
110     DIM XX(PH),YY(PH),ZZ(PH)
120 EH=30:DIME(EH)
130 MH=8: DIMME$(MH)
140     DIMN$(6)
150
1000 REM -1-MENU----------------------
1010 MI=0:GOSUB1250
1020 ME$(0)=" MENU
1030 ME$(1)=" ALAPADATOK
1040 ME$(2)=" PONTOK KOORDINATAI
1050 ME$(3)=" ELEK BEIRASA
1060 ME$(4)=" RAJZOLAS
1070 ME$(5)=" ADATOK TAROLASAGA
1080 ME$(6)=" KILEPES A PROGRAMBOL
1090 FORI=0TOMH:PRINTAT(0,11+I)ME$(I):NEXTI
1100 PRINTAT(0,11+MI)"S"ME$(MI)
1110 GETK$
1120   IFK$<>"C"THEN1130
1130   IFMI>MH-1THEN1160
1140   PRINTAT(0,11+MI)ME$(MI)
1150   MI=MI+1:PRINT"S"ME$(MI)
1160   IFK$<>"S"THEN1200
1170   IFMI<1THEN1200
1180   MI=MI-1:PRINTAT(0,11+MI)"S"ME$(MI)
1190   PRINTME$(MI+1)
1200   IFK$=CHR$(13)THEN 1220
1210 GOTO1110
1220 ONMI+1GOTO1110,2010,3010,4010,5010,6010,7000
1230 GOTO1110
1240
1250 PRINT"S ------------------------------
1260 PRINT"     HAROMDIMENZIOS ABRAZOLAS
1270 PRINT
1280 PRINT ------------------------------
1290 PRINTAT(0,22)"
1300 PRINTAT(0,23)"  (C)BY FEKETE ZOLTAN 1985.
1310 PRINTAT(0,24)"------------------------------
1320 PRINTAT(2,23)ME$(MI)
1330 RETURN
1340
2000 REM -2-ALAPADATOK----------------
2010 FL=0:GOSUB1250
2020 N$(1)="  NEZO KOORDINATAI---X(N)
2030 N$(2)="                    Y(N)
2040 N$(3)="                    Z(N)
2050 N$(4)="  FOPONT KOORDINATAI-X(C)
2060 N$(5)="                    Y(C)
2070 N$(6)="                    Z(C)
2080 FORK=1TO6
2090   PRINTAT(0,14+K)N$(K);   AL(K)
2100 NEXTK
2110 K=1:PRINTAT(0,14+K)"S"N$(K)
2120 AL$=STR$(AL(K)):IFAL(K)=0THENAL$="
2130 GETA$:IFA$=""THEN2130
2140   IFA$="C"THENGOSUB2280
2150   IFA$=" "THENGOSUB2250
2160   IFASC(A$)>44ANDASC(A$)<58THENGOSUB2300
2170   IFA$=CHR$(13)THEN1810
2180 GOTO2130
2190
2200 IFK>5THENRETURN
2210 PRINTAT(0,14+K)N$(K)
2220 K=K+1:PRINTAT(0,14+K)"S"N$(K)
2230 RETURN
2240
2250 IFK<2THENRETURN
2260 PRINTAT(0,14+K)N$(K)
2270 K=K-1:PRINTAT(0,14+K)"S"N$(K)
2280 RETURN
2290
2300 B$=A$
2310 PRINTAT(25,14+K)"S                    "S
2320 GETA$
2330   IFA$=""THEN 2330
2340   IFA$=CHR$(13)THEN2400
2350   IFASC(A$)<45THEN2420
2360   IF ASC(A$)>57 GOTO 2320
2370   IF LEN(B$)<9THENB$=B$+A$
2380   PRINTAT(20,14+K)"S"B$
2390 GOTO 2320
2400 AL(K)=VAL(B$)
2410 PRINT AT(25,14+K)"                    "AL(K)
2420 GOSUB2200
2430 RETURN
2440
3000 REM -3-PONTOK KOORDINATAI----------
3010 FL=0:GOSUB1250
3020 I=1:RESET9020
3030   READX(I):IFX(I)=999THEN3070
3040   READY(I):IFY(I)=999THEN3070
3050   READZ(I):IFZ(I)=999THEN3070
3060 I=I+1: GOTO3030
3070 IM=I-1:GOTO1810
3080
4000 REM -4-ELEK BEIRASA----------------
4010 GOSUB1250
4020 J=1:RESET 9020
4030 READE(J):IFE(J)=999THEN4050
4040 J=J+1: GOTO4030
4050 JM=J-1:GOTO1810
4060
```

— A néző nem lehet az ábrázolandó tárgyon belül.

— A tárgyhoz túl közeli nézőpont felvétele esetében fellépő erős torzulások miatt (fényképezésnél nagy látószögű objektív) célszerű a nézőpontot a 60 fokosnál kisebb látókúp figyelembevételével megadni.

```
5000 REM -5-PERSPEKTIVA SZAMITASA--------
5010 GOSUB1250
5020 PRINTAT(5,10)'1 -- MONITOR VAGY TV '
5030 PRINTAT(5,12)'2 -- PRINTER (MPS-801)'
5040 PRINTAT(5,14)'3 -- PLOTTER (1520)'
5050 GETFL$
5060   IFFL$(>1THENGOSUB5120
5070   IFFL$="1"THENGOSUB5610
5080   IFFL$="2"THENGOSUB5700
5090   IFFL$="3"THENGOSUB5740
5100   IFFL$=CHR$(32)THEN1010
5110 GOTO5050
5120 PRINTAT(2,23)'  TURELEM
5130 IFAL(5)=AL(2)THENAL(5)=AL(5)+AL(5)/1000000
5140 IFAL(5)=AL(2)THENAL(5)=AL(5)+0.00001
5150 GA=ATN(ABS(AL(6)-AL(3))/ABS(AL(5)-AL(2)))
5160 F1=(ATN(ABS(AL(4)-AL(1))/ABS(AL(5)-AL(2))))
5170 S1=SIN(F1):C1=COS(F1)
5180 S2=SIN(GA):C2=COS(GA)
5190 FORI=1TOTM
5200   X1=XX(I)-AL(1)
5210   Y1=YY(I)-AL(2)
5220   XX(I)=-X1*C1+Y1*S1
5230   YY(I)= X1*S1+Y1*C1
5240 NEXTI
5250 FORI=1TOTM
5260   Y1=YY(I)
5270   Z1=ZZ(I)-AL(3)
5280   YY(I)= Y1*C2+Z1*S2
5290   ZZ(I)=-Y1*S2+Z1*C2
5300 NEXTI
5310 NA=1:J=8:J=0
5320 FORI=1TOTM
5330   IFYY(I)<0THENXX(I)=XX(I)/YY(I):YY(I)=ZZ(I)/YY(I)
5340 NEXTI
5350 XN=XX(1):YN=YY(1):XK=XX(1):YK=YY(1)
5360 FORI=2TOTM
5370   IFXX(I)>XNTHENXN=XX(I)
5380   IFYY(I)>YNTHENYN=YY(I)
5390   IFXX(I)<XKTHENXK=XX(I)
5400   IFYY(I)<YKTHENYK=YY(I)
5410 NEXTI
5420 IFXN<>XKANDYN<>YKTHEN5470
5430 PRINTAT(02,23)'HIBAS ADATBEADAS   &RETURN
5440 GETA$
5450   IFA$<>CHR$(13)THEN5440
5460 GOTO1010
5470 NA=310/(XN-XK)   :FL=1
5480 N1=198/(YN-YK)
5490 IFN1<NATHENNA=N1
5500 J=-NA*XK+199-NA*(XN-XK)/2
5510 J=-NA*YK+199-NA*(YN-YK)/2
5520 FORI=1TOTM
5530   X1=XX(I)*NA+U
5540   XX(I)=319-X1
5550   Y1=YY(I)*NA+J
5560   IFAL(5)>AL(2)THENYY(I)=199-Y1:ELSE:YY(I)=Y1
5570 NEXTI
5580 PRINTAT(2,23)' KIJITELI ESZKOZ?
5590 RETURN
5600
5610 HIRES0,1:FF=1
5620 FORJ=1TOJM-1
5630   IFEC(J+1)=0OREC(J)=0THEN5650
5640   LINEXX(EC(J)),YY(EC(J)),XX(EC(J+1)),YY(EC(J+1)),1
5650 NEXTJ
5660 PAUSE1000:*RM
5670 RETURN
5680
5700 IFFF=0THENGOSUB5610
5710 CSET2.COPY.*RM
5720 RETURN
5730
5740 OPEN1,0,1
5750 IFFF=0THENGOSUB5610
5760 CSET2
5770 FORJ=1TOJM
5780 IFEC(J-1)<>0ANDEC(J)<>0THEN5800
5790 PRINT#1,"M";STR$(320-XX(EC(J-1))),STR$(YY(EC(J-1))-60:GOTO5810
5800 PRINT#1,"D";STR$(320-XX(EC(J-1)));STR$(YY(EC(J-1)))
5810 NEXTJ
5820 CLOSE1:*RM
5830 RETURN
5840
6000 REM -6-ADATTAROLAS-----------------
6010 REM RUTIN HELYE
6020 GOTO1010
6030
7000 REM -7-KILEPES A PROGRAMBOL--------
7010 PRINT"Q VEGE" END
7020
8000 REM -8-PONTOK KOORDINATAI----------
8010
8020 DATA   0,  0,  0
8030 DATA   5,  0,  0
8040 DATA   5,  9,  0
8050 DATA   0,  9,  0
8060 DATA   0,  0,  3
8070 DATA   5,  0,  3
8080 DATA   5,  9,  3
8090 DATA   0,  9,  3
8100 DATA -.5, -1,  2.5
8110 DATA 5.5, -1,  2.5
8120 DATA 5.5, 10,  2.5
8130 DATA -.5, 10,  2.5
8140 DATA 2.5, -1,  5.5
8150 DATA 2.5, 10,  5.5
8160 DATA 2.5,  0,  5.5
8170 DATA 2.5,  9,  5.5
8180 DATA 999  - REM PONTOK VEGE
8190
9000 REM -9-ELEK MEGADASA---------------
9010
9020 DATA 1,2,3,4,1,5,15,8,7,16,8,5,8,8,2,8,7,3,8,8,4,8
9030 DATA 18,11,14,12,9,13,10,13,14,8
9040 DATA 999  :REM ELEK VEGE
```

— Távoli nézőpont esetében a perspektív torzulás egyre kisebb (fényképezésnél teleobjektív), igen távoli nézőpont esetében nézetrajzokat, illetve axonometrikus rajzokat kapunk.

A főpont koordinátáinak megadása:

— A főpont megadásával a nézési irányt adjuk meg. A valósághű ábrázolás érdekében célszerű, ha a főpont a tárgy geometriai középpontjához közel esik.

— Amennyiben a néző és a főpont megfelelő koordinátái különböznek, úgy a számított ábra háromiránypontos, ha egy koordináta azonos, úgy kétiránypontos, ha két koordináta azonos, akkor egyiránypontos a perspektív ábra.

3. A pontok koordinátáinak megadása

tetszőleges, célszerűen úgy történik, hogy az ábrázolandó tárgy méretei könnyen leolvashatóak legyenek.

A korlátozott terjedelem miatt a pontok koordinátáit beíró programrészt egy igen rövid rutin helyettesíti, melyben a pontok koordinátáit DATA sorokból READ ciklus olvassa be.

A rutin átírható, a pontok x, y és z koordinátáit kell az X(I), Y(I), Z(I) tömbökbe betöltenünk.

4. Élek megadása

A pontok összekötési sorrendjének megadását szolgálja. 0 számú pont megadása esetén az él nem rajzolódik meg (tollemelés). A programrészt itt is rövid DATA—READ ciklus helyettesíti.

A modul átírása esetén az összekötő pontok sorszámát kell megfelelő sorrendben az E(J) tömbbe töltenünk.

5. Perspektív ábra számítása és rajzolása

A rutin a térbeli koordinátatranszformáció alapadatait, a képernyő méretének megfelelő nagyítás, vízszintes és függőleges eltolás nagyságát számítja, majd az egyes koordináták transzformációja után a képernyőre vagy plotterre történő rajzolás következik.

6. Adatkivitel és adatbehozatal

A pontok koordinátáit, a pontok összekötési sorrendjét, esetleg a perspektív alapadatokat, perspektív koordinátákat, perspektív ábrát háttértárolóra mentő és betöltő modul helye.

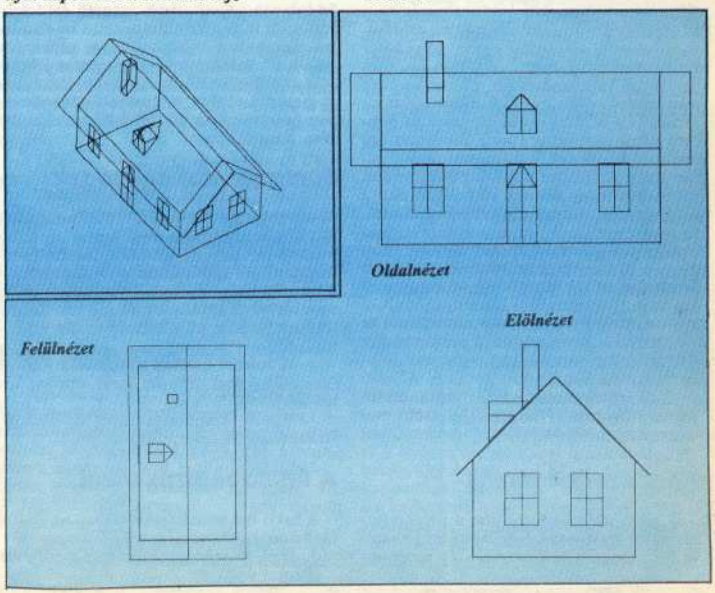
A program átírható és továbbfejleszthető. Működésének megértését és továbbfejlesztését az alábbi szakirodalom tanulmányozása segítheti:

1. W. M. Newman—R. F. Sproull: Interaktív számítógépes grafika. Műszaki Könyvkiadó, 1985.
2. W. Heft: Einführung in CAD mit dem COMMMODORE 64. Data Becker Gmbh, 1985.
3. M. Weber: 3—D—Grafik in Theorie und Praxis. IWT—Verlag, 1984.
4. J. Plate: Abbildung dreidimensionaler Objekte. MC 1986/2.
5. G. Pomaska: Alles darf man nicht zeigen. Chip, 1984/3.
6. G. Pezzi: Studio di Prospettive. MCmicrocomputer n. 31 giugno, 1984.
7. F. Danielowski—A. Pretzsch: Architekturperspektive. VEB Verlag, Berlin, 1969.
8. E. Gull: Perspektivlehre. Verlag für Architektur, Erlenbach—Zürich, 1946.

FEKETE ZOLTÁN

3. ábra. Távoli nézőpont felvételével eredményül kapott axonometrikus rajz

4. ábra. Nézetrajzok előállítása egyirányponttos perspektív rendszer és távoli nézőpont felvételével



Az alkalmazásokban először a központi telepítésű számítógépekkel közeli vagy távoli — például postai távbeszélő-hálózaton keresztüli — vonalkapcsolatban álló, többé vagy kevésbé intelligens terminálok rendszere terjedt el. Ebben a terminálhálózatban az egyes munkahelyekről a felhasználók közösen használták a központi telepítésű, nagy teljesítményű feldolgozó számítógép erőforrásait, tárolt adatállományait, programjait, feldolgozó teljesítményét stb. Az ilyen rendszerekben a terminálok és a központi gép telepítési távolsága több tíz kilométer is lehet, a köztük levő adatátvitel sebességét részben a vonalkapcsolat „minősége" (fizikai jellemzői), részben az adatátviteli készülékek (modemek) paraméterei határozták meg; általában 300—9600 bit/s. Ez a megoldás még nem eredményezett „valódi" számítógépes hálózatot.

A szó szoros értelmében vett számítógépes hálózatban a hálózat egyes gépei, a csomóponti számítógépek — lehetnek azonos vagy különböző típusúak — önálló erőforráskészlettel és intelligenciával rendelkeznek (memória, tárolt információ, önálló feldolgozó teljesítmény stb.), egymás között önállóan forgalmazhatnak (programok távolról történő futtatása, adatcsere stb.), és hozzáférhetnek egymás erőforrásaihoz is.

Az ipar, a tudomány és a gazdaság tevékenységi körében számos esetben van szükség olyan rendszerre, amely viszonylag szűk földrajzi környezeten, például egyetlen vállalaton belül teremt információs kapcsolatot és/vagy végez meghatározott célú feldolgozásokat. Ezek az adottságok, azaz a kis távolság, a jól körülhatárolt intézményi feladatkör, valamint a speciális igények: a nagy adatátviteli sebesség és a megbízhatóság sajátos műszaki megoldásokat kívánnak, amelyeket az ún. helyi hálózatok valósítanak meg.

A helyi hálózatokról kb. 15 éve jelentek meg az első publikációk. A „helyi" jelző elsősorban a hálózat egyes gépeinek és perifériájának viszonylag közeli, általában egy épületen belüli telepítésére utal. A megoldás főleg a rendszer gazdaságossága miatt vált népszerűvé, aminek elsődleges oka a hálózatelérés ötletes műszaki megvalósítása.

A helyi hálózatok fő alkotóelemei:

— a 10 km-nél általában nem nagyobb távolságot átfogó adatátviteli közeg (koaxiális vagy sodrott érpárú kábel),

— a számítógépes munkaállomásokat és a központi, nagy kapacitású erőforrásokat (lemezegység, nyomtató) rendszerbe kapcsoló szerverállomás,

— az egyes munka- és szerverállomásokat az adatátviteli közegre rákapcsoló csatolóegység (LAN board).

Rendszerjellemzők

A *távoli erőforráskapcsolat* a helyi hálózat egyik legfontosabb jellemzője. Lehetővé teszi, hogy a hálózat adat- és programfájljait a hálózat központi telepítésű lemezegységén tárolják (fájl server funkció), amihez a hálózat minden munkaállomása a hálózaton keresztül hozzáférhet. A megoldás növeli a tárolt adatok védelmének hatékonyságát is. Hasonló erőforrás-megosztást tesz lehetővé a „print server" funkció is, melynél a hálózat munkaállomásai közösen használhatják a helyi hálózat központi nyomtatóját.

A *nagy sebességű adatkapcsolat* a helyi hálózat állomásai számára lehetővé teszi a hatékony és gyors adatátvitelt a nagy sebességű adatátviteli csatornán, például koaxiális kábelen keresztül (1—10 Mbps).

A hálózat egyes gépeire telepített *szolgáltatások* hozzáférhetővé tehetők a hálózat többi gépe számára is. Egyes feldolgozások során szükség lehet például a feldolgozandó adatoknak a hálózaton belüli megkeresésére és a feldolgozást végző gépre való átvitelére, vagy a feldolgozás eredményének a hálózati nyomtatón való kinyomtatására.

A terhelések hálózaton belüli átcsoportosíthatósága lehetővé teszi a pillanatnyilag túlterhelődött hálózati számítógép időszakos tehermentesítését, egyes funkciók átirányítását a hálózat más gépeire. Ennek vizsgálatánál természetesen figyelembe kell venni olyan szempontokat is, hogy egy feladatátcsoportosítás mennyire terheli túl a több funkciót „átválló" számítógépet. Az adatátviteli csatorna terhelése például egy meghatározott terhelésen túl már nem növelhető az ezzel együttjáró gyakori adatcsomag-ütközések miatt, így adott esetben fontolóra kell venni a szóban forgó funkciónak ideiglenes letiltását.

A feldolgozó teljesítmény megosztására olyan esetekben kerül sor, amikor egy nagy feldolgozó teljesítményt igénylő feldolgozást csak több részfeldolgozásra bontva lehet végrehajtani a hálózat egyes gépein.

A helyi hálózatok további fontos jellemzője az állomások és az átviteli hálózat igen jó megbízhatósága, melyet jelentős mértékben befolyásol az átviteli hálózat elrendezése, a topológia.

Funkcionális egységek

— A tulajdonképpeni feldolgozásokat végző számítógépek: fájl server állomás (osztott hozzáférést biztosító lemezalrendszer); printer server állomás (osztott hozzáférést biztosító nyomtató rendszer); gateway server (a lokális hálózatot más számítógépes hálózatokkal összekapcsoló számítógép).

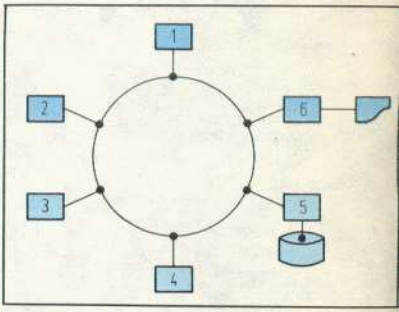
— A lokális hálózat adatátviteli közege (koaxiális kábel, sodrott érpár, optikai kábel stb.).

— Egyéb kommunikációs hálózatkapcsolat (telefon, telex, videotex stb.).

A fejlődés szakaszai

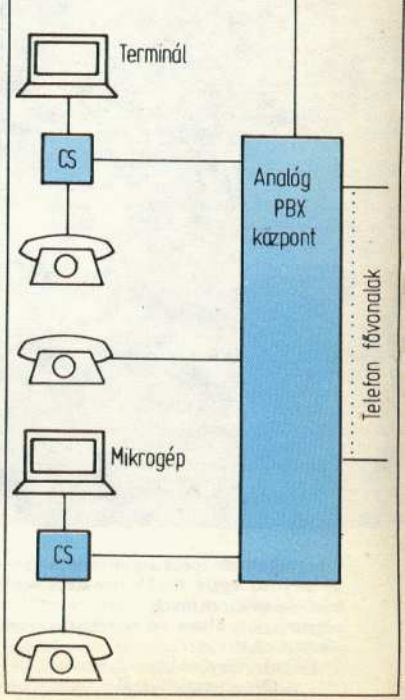
A helyi hálózatok fejlődésében az alábbi funkcionális fázisok figyelhetők meg:

1. fázis. A számítógépek és a perifériák



1. ábra. Az általános célú helyi hálózat elvi vázlata
1—4: Munkaállomások (lehet terminál, PC stb.)
5: Osztott hozzáférést biztosító lemez-alrendszer
6: Osztott hozzáférést biztosító nyomtató-alrendszer

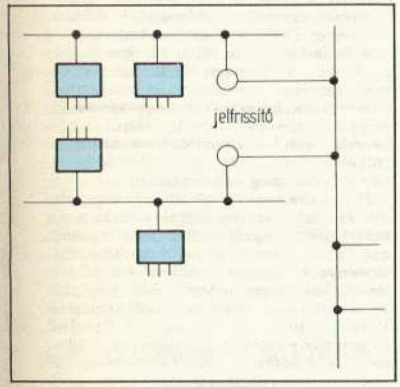
2. ábra. Analóg üzemmódú PBX helyi hálózat elvi vázlata



közti adatátviteli teljesítmény növelése — például koaxiális kábeles rendszer alkalmazása, szemben a hagyományos rendszerben alkalmazott modemes átvitellel. Ennek eredményeként a csatorna átviteli sebesség felső határa 9600 bps-ról 1—10 Mbps-ra növekedett.

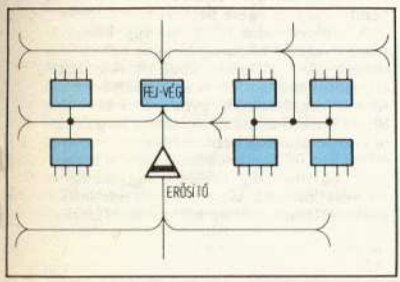
2. fázis. Az adatátviteli hálózatra csatlakoztatható egységek (PC-k) választékának rugalmas bővíthetősége: újabb gépek bekapcsolása a hálózatba, viszonylag egyszerű módon.

3. fázis. Funkciómegosztás a hálózati struktúrán belül, hálózatok közötti kommunikáció (gateway) megvalósítása.



3. ábra. Alapsávú helyi hálózat elvi vázlata

4. ábra. Széles sávú helyi hálózat elvi vázlata



4. fázis. Az adatátvitellel párhuzamosan megvalósított kép- és hangátvitel (széles sávú rendszerek). Komplex szolgáltatást nyújtó integrált információs rendszerek létrehozása.

Adatátviteli közeg

A helyi hálózatok adatátviteli közege (kábel) lehet sodrott érpárú, mellyel általában néhányszor tíz állomás kapcsolható össze maximum 1—2 Mbps sebességű rendszerré. Ez a közeg érzékeny a zajra, áthallásra, beruházási költségei viszont az összes megoldás között a legalacsonyabbak.

Koaxiális kábellel már akár több száz állomás is összekapcsolható, maximum 10 Mbps-os sebességű rendszerré. Koaxiális kábellel működő helyi hálózat lehet alapsávú (baseband), mellyel egyidőben csak egyetlen csatornán létesíthető átvitel, vagy széles sávú (broadband), mellyel egyidőben több csatornán keresztül is létesíthető átvitel. Az adatátvitellel egyidőben hang- és képátvitelre is van mód.

A jövő átviteli közege az optikai kábel, melynek fő előnye a kis súly és huzalkeresztmetszet, a kiváló zaj- és áthallásvédelem, a nagy átviteli sávszélesség. Ilyen kábellel jelenleg csak néhányszor tíz állomás kapcsolható össze, maximum 10 Mbps-os rendszerré, kb. 1 km-es távolságon belül. A megoldás szélesebb körben való elterjedését ma még gátolja az optikai kábel meglehetősen magas ára.

Hálózattípusok

Az *általános célú helyi hálózatok (1. ábra)* a legelterjedtebbek. Nagy központi számítógépeket, mini- és mikrogépeket, képernyős terminálokat és egyéb perifériákat kapcsolnak össze velük, rendszerint sín- vagy fastruktúrájú hálózattá, koaxiális vagy sodrott érpárú kábel segítségével. Átviteli sebességük 1—10 Mbps körüli. A hálózatba kapcsolható készülékek száma több száz lehet. Adat, hang és képi információ is továbbítható.

A *nagy sebességű helyi hálózatot* rendszerint nagy számítóközpontokban alkalmazzák a központi számítógép és a nagy sebességű perifériák közötti minél gyorsabb adatátvitel megvalósítására. Az ilyen rendszerek átviteli sebessége akár 50 Mbps is lehet, az összekapcsolható eszközök száma csekély.

A fenti két hálózat csomagkapcsolt típusú.

A *telefonhálózaton keresztül* vonalkapcsolásos választással *működő helyi hálózatokat* a szakemberek PBX (Public Branch Exchange) néven ismerik. A hálózat az intézményeknél már meglevő helyi telefonhálózatra és telefonközpontokra épül, rajta beszéd és adat egyaránt továbbítható.

PBX hálózatok

1983-ban az USA-ban az ilyen elven működő rendszereknek több mint 300-féle változatát tartották nyilván. A változatok nagy számát az magyarázza, hogy ezek a rendszerek a már meglevő és igen sokféle megoldás továbbfejlesztéseként alakultak ki. A leggyakoribb típusokat néhány szóval bemutatjuk.

Az *analóg hálózatokban (2. ábra)* az átvitelt modemmel valósítják meg, a telefonhálózathoz olyan egységen keresztül csatlakoztatják a munkahelyeket, amely biztosítja a telefonkészülék csatlakoztatását is. A csatlakoztató egységen kapcsolóval választható a beszéd (telefon) vagy az adatátviteli (terminál) üzemmód, egyidőben ugyanis vagy csak beszéd, vagy csak adatátvitel hajtható végre.

A *hibrid hálózatokban* a beszéd- és az adatkapcsolatot egymástól független alközpontok vezérlik. A készülékeknek a hálózathoz való csatlakoztatását itt is egy csatlakoztató egység biztosítja, mely a beszédátvitelt alapsávon (300 Hz—4 kHz), az adatátvitelt a 36—40, illetve a 72—80 kHz sávtartományban valósítja meg. Annak ellenére, hogy az adat- és a beszédátvitel közös érpáron történik, a kétféle átvitel független egymástól, tehát egyidőben is megvalósítható.

Az *analóg vonallal digitális központhoz kapcsolódó hálózatokban* a központ maga digitális elven működik, vonalkapcsolatait analóg-digitális átalakítók biztosítják. Ez a rendszer az analógnál megbízhatóbb, kapcsolási sebessége nagyobb, méretei kisebbek, költségei alacsonyabbak.

A *digitális vonallal digitális központhoz kapcsolódó hálózatok* a telefonkészülékben digitalizálják (kódolják) a beszédjeleket, majd ezt viszik át a hálózaton keresztül a digitális központba.

Az *egyedi PBX hálózatok* kapcsolóközpontja két szektorra oszlik, s ezek vagy csak telefon-, vagy csak adatkapcsolatot valósítanak meg. Vonalai közül egy a telefonösszeköttetésre, kettő az adatösszeköttetésre szolgál. A telefon és az adatkapcsolat egymástól függetlenül működik. Ez a típus a leglassúbb az összes között, ezért csupán szerényebb igények kielégítésére alkalmas. Az átvitt üzenetek hossza tetszőleges, a kapcsolatfelépítés lassú, ami egyúttal meghatározza alkalmazási körét is. Beruházási költségigénye — tekintettel a már minden intézménynél meglevő helyi telefonhálózatra — viszonylag alacsony.

Működési frekvencia

A helyi hálózatok működésük frekvenciatartományа szerint lehetnek alapsávban működő (baseband) vagy frekvenciaáthelyezéssel működő, ún. széles sávú (broadband) rendszerek.

Az *alapsávú rendszerekkel (3. ábra)* 1 km körüli távolság hidalható át. Az átviteli közeg 50 ohmos koaxiális kábel, amely kevésbé érzékeny a csatlakozások okozta reflexiókra és a zajra. Az átvitel sebessége 10 Mbps, a csatlakoztató munkahelyek száma kb. 100. Alacsonyabb sebességű, olcsóbb hálózatokban átviteli közegként használható a sodrott érpárú kábel is. Az átviteli sebesség ekkor csak kb. 1 Mbps, a rendszerbe kapcsolható állomások száma néhányszor tíz, zajérzékenysége viszont alig rosszabb a koaxiális kábelt használó rendszerekénél.

A *széles sávú hálózatok (4. ábra)* analóg jelek frekvenciaosztásos átvitelével működ-

nek, rajtuk az adatátvitellel egyidőben kép és hang is továbbítható. Rendszerint sín- vagy fastruktúrájúak, az átvitel egy- vagy kétirányú, az áthidalható távolság több tíz km, a rendszerbe kapcsolható állomások száma akár több száz is lehet.

Kétirányú átvitelnél az egyik megoldás szerint egy adási és egy vételi adatútra van szükség, melyek a hálózat egy pontján, az ún. „fejvég"-en (head-end) csatlakoznak egymáshoz. A másik megoldásnál csak egy adatút van, itt viszont az adási, illetve vétel irányú vivőfrekvenciák különböznek egymástól. Az ilyen megoldás „fejvég"-e frekvenciaváltóként működik. A legismertebb rendszerekben az adási frekvenciasáv 5–116 MHz, a vételi 168–300 MHz tartományba esik.

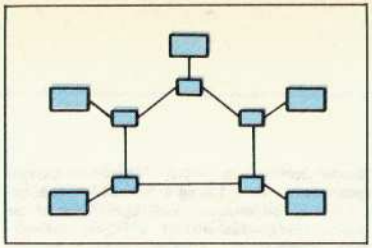
Ha egy széles sávú rendszert egynél több csatornásra készítenek, a többi csatornán az adatátvitellel párhuzamosan kép- (tv) és hangátvitel is megvalósítható.

Hálózati elrendezések

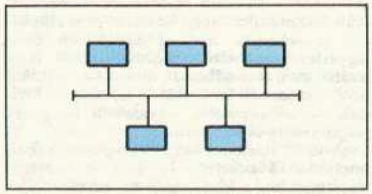
Hálózati elrendezésen a hálózat állomásainak egymással való összekapcsolási sémáját értik. A hálózati elrendezés befolyással van a hálózat paramétereire, például a megbízhatóságra.

A *gyűrűs hálózat (5. ábra)* igen érzékeny az átviteli közeg hibájára, mivel rajta az állomások egymás után, sorosan vannak felfűzve, ezért ha valahol a kábel megsérül — megszakad vagy zárlatos lesz —, a teljes hálózat működésképtelenné válik. Az ilyen típusú hálózat úgy működik, hogy az üzenetküldő állomás ráteszi a gyűrűre forgalmazandó üzenetét, a sorban őt követő állomás megvizsgálja, hogy az üzenet neki szól-e, és ha igen, beolvassa annak tartalmát. Ha nem neki szól, továbbítja az üzenetet a következő állomásnak. Az üzenet adott irányban halad végig a gyűrűn mindaddig, amíg meg nem találja a címzett állomást, amely kiolvassa az üzenet tartalmát, és a vételt nyugtázva beállítja és visszaküldi a „sikeresen kiolvasva" jelzést a forgalmazó állomásnak. Léteznek olyan rendszerek is, melyekben a gyűrűben egyszerre több üzenet is keringhet.

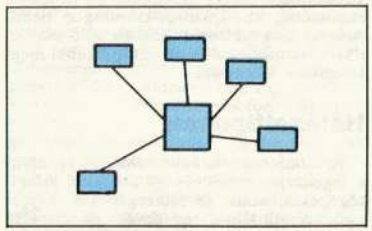
A *sín (busz) hálózatban (6. ábra)* egyszerre csak egy üzenet továbbítható. A hálózat működtethető központi vezérléssel is, ilyenkor a forrásállomás által küldött adatcsomag először a központba továbbítódik, majd a fogadóállomáshoz. Az egyidőben forgalmazni kívánó forrásállomások „versenyeznek" a hálózathasználat jogáért.



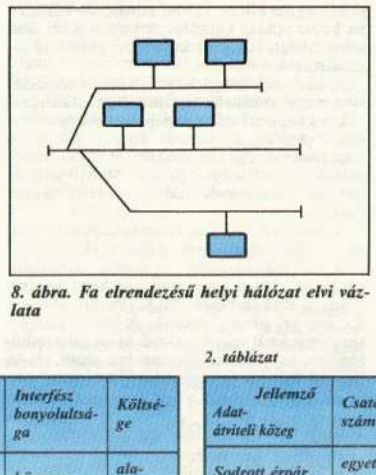
5. ábra. Gyűrűs elrendezésű helyi hálózat elvi vázlata



6. ábra. Sín elrendezésű helyi hálózat elvi vázlata



7. ábra. Csillag elrendezésű helyi hálózat elvi vázlata



8. ábra. Fa elrendezésű helyi hálózat elvi vázlata

A hálózathasználat jogát többféleképpen lehet biztosítani: *központilag*, amikor a központi állomás egy-egy időtartamra egymás utáni adási joggal ruházza fel a hálózat állomásait (polling technika), és ha az adásra feljogosított állomás forgalmazni akar, a számára kijelölt időtartamra adásra lefoglalhatja a sínt; *időosztásos módszerrel* (TDMA = Time Division Multiple Access) és *frekvenciaosztásos módszerrel* (FDMA = Frequency Division Multiple Access).

A sín megszerzése iránti kérelem indítása, illetve a megszerzett sín használatba vétele történhet tetszőleges időpillanatban (random) is. Ez azzal jár, hogy az egyidőben kezdeményezett forgalmazások „ütközhetnek", aminek következtében az „ütköző" üzenet elvész, és egy későbbi időpontban az adást meg kell ismételni.

Ha a sínre nagyon sok állomás kapcsolódik és ezek gyakran forgalmaznak, a sín túlterhelődik, ugrásszerűen megszaporodnak az „ütközések", emiatt a rendszer teljesítménye erőteljesen csökken. Ezt elkerülendő, különleges módszereket dolgoztak ki az ütközések számának csökkentésére. Ilyenek a sínforgalomba való „befigyelés" (forgalmaz-e valaki már a sínen) és az adásismétlés időzítésének adott algoritmus szerinti vezérlése (például ütközés esetén az első ismétlésre t idő múlva, a másodikra 1,5 t múlva stb. kerül sor).

A sínelrendezésű hálózat egyszerűen bővíthető, és az egyes állomások hibája nem bénítja meg az egész hálózat működését.

A *csillaghálózat (7. ábra)* egy központi hálózatvezérlő állomásból és az ennek alárendelt munkaállomásokból áll. A munkaállomások a központi vezérlőállomáson keresztül forgalmaznak egymással, a hívó és a hívott közötti adatkapcsolatot a forgalmazni kívánó állomás kezdeményezésére a központ építi fel. Az elrendezés előnye az egyszerű bővíthetőség, továbbá hogy az egyes munkaállomások kiesése nem teszi működésképtelenné a teljes hálózatot. Hátránya

1. táblázat

Jellemző / Elrendezés	Rugalmasság	Bővíthetőség	Megbízhatóság	Interfész bonyolultsága	Költsége
Sín	nagy	nagy	nagy	közepes	alacsony
Gyűrű	közepes	közepes	nagy	alacsony	közepes
Csillag	alacsony	nagy	közepes	alacsony	nagy
Fa	nagy	nagy	nagy	közepes	alacsony

2. táblázat

Jellemző / Adatátviteli közeg	Csatornaszám	Karbantartás	Állomásszám	Kábelhossz	Zavarvédelem
Sodrott érpár	egyetlen (adat/hang)	nehézkes	kevés	néhány mérföld	gyenge
Koaxiális alapsáv	egyetlen (adat)	igen egyszerű	sok	néhány mérföld	közepes
széles sáv	több száz (adat, hang, kép)	könnyű	nagyon sok	40 mérföld	kiváló
Optikai kábel	több száz	nagyon nehéz	kevés	korlátlan	kiváló

viszont, hogy a hálózatvezérlő állomás hibája az egész hálózat működését megbénítja.

A *fa-hálózat* csillagelrendezéseket kapcsol össze hierarchikus felépítésű és vezérlésű hálózattá *(8. ábra)*. A struktúra előnye egyszerű bővíthetősége, hátránya viszont, hogy a magasabb hierarchiaszintekhez tartozó útvonalak kiesésével önálló hálózatokká esik szét, továbbá, hogy egyes adatutak túlterheltsége (bedugulása) esetén a hozzá tartozó hálózati részekről elküldött üzenetek késleltetése megnő.

A leggyakrabban használt hálózati elrendezések összehasonlítása az *1. táblázaton* látható.

A felsoroltakon kívül léteznek további hálózati elrendezések is — főleg kombinált megoldások —, ezek azonban különböző okok miatt kevésbé terjedtek el.

Fel kell hívni a figyelmet arra, hogy a hálózat topológiája és az alkalmazott kábeltípus nem választható meg egymástól függetlenül. A sintopológiához például sodrott érpárú és koaxiális kábel is használható, míg a fastruktúrájú hálózathoz inkább széles sávú koaxiális kábelt, a gyűrűs hálózathoz sodrott érpárú, alapsávú koaxiális vagy optikai kábelt, csillaghálózathoz leginkább sodrott érpárt alkalmaznak.

A különböző adatátviteli közegek hatását a velük összekapcsolt rendszerekre a *2. táblázat* szemlélteti.

Hálózatelérési módszerek

A helyi hálózatok másik lényeges jellemzője a hálózathoz való hozzáférés elve, azaz, hogy milyen módszer szerint érik el az egyes munkaállomások a csatornát, és hogyan forgalmaznak azon.

A hálózat-hozzáférést befolyásolja a csatornakapacitás (Mbps), a közösen használt átviteli kábel, a címfigyelés (a hálózati állomások érzékelik a csatorna forgalmát, de csak a címzett állomás olvassa le róla az üzenetet) és a hálózati elrendezés (topológia).

Decentralizált vezérlésű gyűrűs hálózat esetén a következő hálózat-hozzáférési módszerek terjedtek el: vezérelt jelzéses (token passing), réselt gyűrűs (slotted ring), pufferbeszúrásos (puffer insertion).

A *vezérelt jelzéses hálózatban* az egyes állomások egymás után sorosan kapcsolódnak a hálózatra, ily módon alakítva ki a gyűrűs hálózati struktúrát. A hálózati kábel lehet sodrott érpár vagy koaxiális. A hálózatban a jelzés (token) állomásról állomásra körbejár.

Ha egyik állomás sem kíván forgalmazni, a hálózat nyugalomban van, és a szabad vezérlőjelbe (token) egyik állomás se tesz bele csomagot (üzenetet). Ha valamely állomás forgalmazni kíván, megvárja, amíg a szabad vezérlőjel hozzá ér, majd lefoglalja (foglaltsági jelzés beállítása), belerakja üzenetét (adat, feladó és címzett címe), és a hálózatra teszi. A jel haladási iránya szerint következő állomás érzékeli a „foglalt" jelzést és benne az üzenetet, leolvassa annak címrészét és összehasonlítja a sajátjával, ha megegyezik, kiolvassa az adatrészt is, és beállítja a „sikeres forgalmazás"-t jelző nyugta bitet, és továbbengedi azt a gyűrűben. Ezt a nyugta üzenetet a forrásállomás „veszi", és a „sikeres átvitel" jelzésből tudja, hogy forgalmazása sikeres volt, így azt nem kell megismételni. Mindig a forrásállomás távolítja el a hálózatból a feleslegessé vált nyugta üzenetét és tesz helyette egy szabad jelzést a hálózatba, mellyel elsőként a gyűrűben rá következő állomás forgalmazhat, ha kíván. Ha nem, úgy a szabad jelzés — és ezzel a forgalmazás joga — az őt követő állomásra száll át.

A vezérelt jelzéses gyűrűs hálózatban megvalósítható a többszintű prioritás, azaz a magasabb prioritású állomások felfüggeszthetik az alacsony prioritású állomások adáshoz jutási jogát.

Az állomás addig nem generál új adási jogot, ameddig saját csomagját nyugta formájában vissza nem kapta. Ezt a rendszert nevezik egyetlen adási jogú rendszernek, és e megoldással a rendszer megbízhatósága nagymértékben növelhető, ha a forgalmazni szándékozó állomás előbb meggyőződik a hálózat hibátlanságáról, és csak azután kezd el adni.

A *réselt gyűrűs hálózateltérés* esetében a gyűrűben előírt hosszúságú üzenet befogadására szolgáló „rések" keringenek, amelyek lehetnek vagy „üres" vagy „tele" rések; ezt egy állapotbit jelzi a résen belül. Az az állomás, amely forgalmazni kíván, megvárja, hogy egy rés hozzá érkezzen, majd megvizsgálja, hogy üres-e a rés, és ha igen, „beleteszi" adott hosszúságúra tördelt üzenetét. A most már üzenettel feltöltött rés címzését az állomások sorban megvizsgálják, nekik szól-e az üzenet. Ha igen, kiolvassák annak tartalmát, beállítják a sikeres vételt nyugtázó bitet, és továbbengedik a kiolvasott, de még nem üres rést a csomagot feladó állomás felé. A rés kiürítése (szabaddá tétele) a csomagot feladó állomás feladata. A kiürített rést ezt követően a sorrendben következő állomásnak kell felajánlani, ezzel megakadályozva, hogy egy állomás monopolizálja az adási jogot, kizárva más állomásokat a forgalmazás lehetőségéből. A hálózatban egyidőben keringetett rések számát a hálózat saját késleltetése korlátozza.

A *pufferbeszúrásos hálózatelérés* változó hosszúságú üzenetek átvitelét teszi lehetővé. Az üzeneteket átmeneti tárolóban tárolják. Az egyes munkaállomások interfészei az üzenetek átmeneti tárolására szolgáló puffereket tartalmaznak. Az adó állomás a gyűrűn keringő két üzenet közé szúrja be a sajátját, erre az időre a második üzenet a pufferben tárolódik. Az eljárás segítségével az átlagos üzenetátviteli késleltetés csökkenthető. A sikeresen vett üzeneteket a vevő oldal „távolítja" el a hálózatból.

CSEH KÁLMÁN

COMMODORE 64

Sztringváltozóban elhelyezettt összefüggést kiértékelő szubrutin

A program a kiértékelendő kifejezést tagokra, a tagokat tényezőkre bontja. Kiértékeli a tényezőket, elvégzi a szorzási, osztási műveleteket — azaz kiértékeli a tagokat —, és azokat az E változóba összegzi.

A tag és tényező fogalmakon a következők értendők. *Tagok* a kifejezés azon részei, melyek "+" illetve "−" műveleti jelek között helyezkednek el. A tagok "*" és "/" műveleti jeleket, általunk definiált függvényjeleket és számkonstansokat tartalmazhatnak.

Tényezők a tagok azon részei, melyek "*", illetve "/" műveleti jelek között találhatók. A tényezők számkonstansokat és az általunk definiált függvényjeleket tartalmazhatják.

Fő részei

100—191	Magyarázó feliratok képernyőre
200	Értékadás D$ sztringnek, majd sztring lezárása "+" karakterrel
216	Pointerek, flagek lenullázása, D$ hosszának vizsgálata, tárolása H változóban
220	Ha a H=1 feltétel igaz, akkor csak a RETURN billentyű kódját — chr$(13) — tartalmazza a D$ sztring. Az eredmény definíciószerűen =Ø. Az 530-as sornál kezdődő programrész ezt kijelzi
230—520	A D$ sztringben tárolt kifejezés tagokra, tényezőkre, függvényekre bontása. A kifejezés kiértékelése
530—550	Eredmény kijelzése
560—570	Várakozás. RUN-nal változók törlése, program újraindítása
600—660	Hibaüzenet megjelenítése a képernyő alsó részén.

A kiértékelendő kifejezés a D$ sztring-

ben található. D$ sztringnek a 200-as sorban szereplő INPUT utasítás ad értéket. Az értékadás után D$ sztringhez hozzáadunk egy "+" karaktert, mely az utolsó tag végét jelzi a 260-as sor számára.

A 230—520-as sorok a működési elvnek megfelelően épülnek fel. A 230—300 szubrutin végzi a tagok leválasztását. Megkeresi M-től H (D$ hossza)-ig a +, illetve a − előjeleket, ezt s2-ben, majd s1-ben tárolja. A keresés úgy történik, hogy D$ sztringet karakterenként vizsgálja.

A tagot a D1$ sztringben tárolja, majd hívja a 320—440-es sorokban található szubrutint, amely a D1$-ben tárolt tag kiértékelését végzi. H2 a D1$ hossza. A D1$-ben tárolt tagot ez a szubrutin nem karakterenként vizsgálja, hanem kihasználja a val( ) függvény tulajdonságát, vagyis azt, hogy a sztring számkarakternek megfelelő részét adja át valós számként E1-nek.

Ha E1 = val(D1$) értékadás után E1 = 0 és h2 < > 0, akkor D1$ kifejezés nem számkarakterrel kezdődik, tehát meg kell vizsgálni, hogy a kezdő karakter egyezik-e valamelyik általunk definiált függvényjellel.

Ezt a 460—501-es sorokban található szubrutin végzi, de hívása előtt Q$ sztringváltozóba helyezzük a vizsgálandó karaktert, L3 és L2 pedig azt tárolja, hogy a D1$ sztringben mi a következő kiértékelendő tényező pozíciója.

A függvénykiértékelést a 460—520-as programrész végzi. Itt az általunk definiált függvények argumentuma csak számkonstans lehet, mivel nincs a programban olyan szubrutin, amely elvégezné a függvényargumentumok kiértékelését.

Példák

1. A kifejezés:

−3.3*s12+23.5*s25*t30−29.34

A 200-as sorban szereplő INPUT utasítás a fenti értéket adja D$ sztringnek. Ehhez (D$ = D$ + "+") egy karaktert hozzáadunk. Így az általunk begépelt kifejezés alakja D$-ben:

−3.3*s12+23.5*s25*t30−29.34+

1. 2. 3.

A kifejezés 3 tagból áll. Az első tag 2 tényezőt tartalmaz: egy konstanst és egy kiértékelendő függvényt (s12 = > sin(12)). A második tag 3 tényezőt tartalmaz: egy konstanst és két kiértékelendő függvényt s25 = > sin(25), t30 = > tan(30). A harmadik tag egy számkonstans.

A programlistából kitűnik, hogy a szögfüggvények definiálása oly módon történt, hogy a kifejezés megadásakor beírt szögfüggvények argumentumai szögfokokat jelentenek. Mivel a szögfüggvények definícióihoz az alap BASIC szögfüggvényeit használtam, és mint ismeretes, e függvényargumentumokat radiánként értelmezi az interpreter, a fokértékeket radiánra kellett átszámítani. Ezt a 465-ös sorban a P2 változó tartalmazza.

2. A kifejezés: 2*3*6.2

A 2*3*6.2 kifejezés D$-be kerül, majd a "+" jel hozzáadása után a 2*3*6.2+ alakot kapja. Itt a 2*3*6.2 kiértékelendő kifejezés egyetlen tagból áll, a tagnak 3 db konstans tényezője van.

A program a 470—500-as sorok megváltoztatásával alakítható át egyedi igények esetén. Természetesen más, az itt bemutatottnál összetettebb függvények is definiálhatók.

Új függvény definiálásának szabályai

1. A definiált függvények jele egy karakter hosszúságú lehet, és nem használható numerikus karakter.

2. Egyetlen definiálandó függvényt sem jelölhetünk már felhasznált függvényjellel.

3. Ha a definiált függvény argumentumát szögfokokban kívánjuk megadni, és ez az argumentum a kiértékelés során egyben szögfüggvény argumentuma is, akkor a szögfüggvénynek P2-vel kell számolnia, egyéb függvényargumentumoknak P3-mal. Definiálható olyan összetett függvény is, amely P2-vel és P3-mal egyaránt számol.

Példa függvénydefiniálásra

501 if q$"A"thenp2 = p3!2*PI/4

A definiált függvényre "A3Ø" formában hivatkozhatunk, és egy 3Ø egység átmérőjű kör területét számítja ki.

Ha az 5Ø1-es sort például a következőképpen definiáljuk:

5Ø1 if q$ = "A"thenp2 = p3!2*PI/4*15Ø,

akkor az "A3Ø" hivatkozás egy 15Ø egység magasságú, 3Ø egység átmérőjű henger térfogatát számítja ki.

Végül megemlítjük az összefüggés-kiértékelő szubrutin hiányosságait: a definiált függvények argumentuma csak számkonstans lehet; a kifejezések megadásakor zárójelek nem használhatók; "kiakadás" elleni védelme nem százszázalékos. Természetesen további programrészek megírásával e hiányosságok megszüntethetők.

SCHELL FERENC

```
100 V=53280:POKEV,6:POKEV+1,6:PRINTCHR$(142)CHR$(8)
110 PRINT"■■■HASZNALHATO■"CHR$(13)"     JELEK:"
120 PRINT"■▲■■■■■■■■■■■■■■+ => OSSZEADAS"
130 PRINT"■■■■■■■■■■■■■■- => KIVONAS"
140 PRINT"                * => SZORZAS"
150 PRINT"                / => OSZTAS"
160 PRINT"■■■               S => SIN PL: S25=>SIN(25)"    "
170 PRINT"                C => COS PL: C32=>COS(32)"    "
180 PRINT"                T => TAN PL: T67=>TAN(67)"    "
190 PRINT"                L => LN  PL: L76=>LN(76)"    "
191 REM  ***    BOVITES    ***
192 :
200 INPUT"■■■OSSZEFUGGES■";D$:D$=D$+"+"
215 :
216 H=LEN(D$):L1=0:L2=0:L3=0:H1=0:H2=0:H3=0
220 IFH=1THEND$="":GOTO530
225 :
230 E=0:S1=1:M=1:L1=1:IFLEFT$(D$,1)="-"THENS1=-1:L1=2:M=2
235 IFLEFT$(D$,1)="+"THENL1=2:M=2
240 FORI=MTOH:A$=MID$(D$,I,1)
250 IFA$="-"THENS2=-1:GOTO280
260 IFA$="+"THENS2=1:GOTO280
270 GOTO300
275 :
280 M1=I-D1:D1$=MID$(D$,L1,M1-L1):GOSUB320
290 E=E+S1*E1:S1=S2:L1=M1+1
295 :
300 NEXT
305 :
310 GOTO530
315 :
320 H2=LEN(D1$):E1=VAL(D1$):L2=LEN(STR$(E1)):L3=L2:Z=0
330 IFH2=0THEN600:REM * HIBAUZENET *
340 IFE1=0THEN0$=LEFT$(D1$,1):L3=1:L2=1:GOSUB460:E1=P2:L3=L2
350 Z=1
360 IFL2-1=H2THENRETURN
365 IFL3>H2THEN600
370 P2=VAL(RIGHT$(D1$,H2-L3))
380 IFP2=0THEN0$=MID$(D1$,L3+Z,1):GOSUB460:Z=1:GOTO400
390 L2=LEN(STR$(P2))
400 A$=MID$(D1$,L3,1)
410 IFA$<>"*"ANDA$<>"/"THEN600:REM * HIBAUZENET *
420 IFA$="*"THENE1=E1*P2:GOTO440
430 E1=E1/P2
440 L3=L3+L2:IFL3-1=H2THENRETURN
445 GOTO370
450 GOTO600
455 :
460 IFH2-L3-Z<0THEN600
465 P2=VAL(RIGHT$(D1$,H2-L3-Z)):L2=LEN(STR$(P2))+1:P3=P2:P2=P2*π/180
470 IF0$="S"THENP2=SIN(P2):GOTO520
480 IF0$="C"THENP2=COS(P2):GOTO520
490 IF0$="T"THENP2=TAN(P2):GOTO520
500 IF0$="L"THENP2=LOG(P3):GOTO520
501 REM  ***    BOVITES    ***
502 :
510 GOTO600:REM * HIBAUZENET *
520 RETURN
525 :
530 IFE>=0THEND$=RIGHT$(STR$(E),LEN(STR$(E))-1):GOTO550
540 D$=STR$(E)
550 PRINT"■■■■■ EREDMENY: ■"D$
560 GETA$:IFA$=""THEN560
570 RUN
600 REM ** HIBA **
610 H$="■■ HIBAS OSSZEFUGGES !":GOSUB650
620 GETA$:IFA$=""THEN620
630 RUN
650 POKE214,22:POKE211,1:SYS58732
660 PRINTLEFT$(H$+"                                        ",39):RETURN
```

Egyszerűbben assemblerül

Lebecsülte Lindeisz László a Floppykezelés Kernal-rutinok segítségével c. hozzászólásában a HELP+ és a PROFIMAT fordító lehetőségeit (1986/3. szám). A Frey Judit azonos című cikkéhez (1985/4. szám) kapcsolódó írásában megkérdőjelezi a címkecím alsó és felső bájtját értelmező és „használhatóvá tevő" operandus létezését. Ezt a többi között azzal támasztja alá, hogy csupán az Advanced Machine Code Programming for the Commodore 64 (Granada Technical Books) című könyvben talált utalást a MIKRO—64 assemblerre, amely szerinte csak *talán* ismeri a fenti operandust.

Én szintén Frey Judit cikke kapcsán jutottam el a problémához. A HELP+-ról szóló leírást Erdős Iván: Commodore 64 assembly című könyvében találtam meg, a PROFIMAT-féle alkalmazást pedig próbálgatással okoskodtam ki, ahogyan az alábbi két példa is mutatja.

Mindkét példaprogram nyolc csillagot rajzol a képernyőre. Az LI szubrutin használatához az UZ címe alsó bájtját az A regiszterbe, a felső bájtját pedig az Y regiszterbe kell tenni. A . BYTE direktíva decimális számot épít be a tárgyprogramba, a .TEXT pedig a karakterek ASCII kódjait. A szöveg végét zérus bájt jelenti. Az első példaprogram a PROFIMAT, a második a HELP+ fordítóját használja. Ily módon nem okozhat gondot a SETNAM rutin használata sem, mivel mindkét assembler megengedi a címkék értékének (címének) kivonását, sőt összeadását is. Ezért nagyon egyszerűen kiszámíthatjuk például a fájlnév hosszát.

Így használhatók eredeti formájukban — kis változtatással — Frey Judit részprogramjai, és a programunk is relokálható lesz.

DOSZTÁNYI ZSOLT

```
C000                    100             *=$C000
C000 4C 0C C0           110             JMP KEZD
MB1E                    120 LI          =$MB1E
C003 2A 2A 2A           130 UZ          .TEXT"****"
C007 2A 2A 2A           140             .TEXT"****"
C00B 00                 150             .BYTE 00
C00C A9 03              160 KEZD        LDA #UZ/
C00E A0 C0              170             LDY #UZ?
C010 20 1E MB           180             JSR LI
C013 60                 190             RTS
C014                    200             .END

ZEILEN:11   SYMBOLE:3   FEHLER:0

KEZD  =C00C   LI   =MB1E   UZ   =C003
```

1. példa

2. példa

```
PELDA-M

Z
240   C000                              .OPT P3,00
120   C000                              *=  $C000
130   C000 4C 0C C0                     JMP  KEZD
140   C003                 LI           =    $MB1E
150   C003 2A 2A           UZ           .BYTE42,42
160   C005 2A 2A                        .BYTE42,42
170   C007 2A 2A                        .BYTE42,42
180   C009 2A 2A                        .BYTE42,42
190   C00B 00                           .BYTE00
200   C00C A9 03    KEZD                LDA  #<UZ
210   C00E A0 C0                        LDY  #>UZ
220   C010 20 1E MB                     JSR  LI
230   C013 60                           RTS
```

```
100 ;┌──────────────────────┐
100 ;!INPUT+  UTASITAS C-64-RE!
100 ;!    BY KEINDL LASZLO    !
100 ;!(C) 1986.03.06.TATABANYA!
100 ;└──────────────────────┘
150         *=C000
160 BASVEC  =$0308  ;776 RENDSZERMUTATO
170 CHRGET  =$73    ;CHRGET RUTIN (115)
180 CHRGOT  =$79    ;CHROOT RUTIN
190 RINPUT  =$ABBF  ;REGI INPUT RUTIN
200 RINTER  =$A7E7  ;INTERPRETER
210 PLUSZ   =$AA    ;(+) JEL KODJA
220 INTER   =$A7AE  ;INTERPRETER
230 INPUT   =$85    ;INPUT TOKEN
240         LDA #VIZSG<  ;CIMBETOLTES
250         LDY #VIZSG>
260         STA BASVEC   ;A RENDSZER-
270         STA BASVEC+1 ;MUTATOBA
280         RTS
290         NOP
300         NOP
310         NOP
320 VIZSG   JSR CHRGET
330         CMP #INPUT
340         BEQ TALAL
350         JSR CHRGOT
360         JMP RINTER
370 TALAL   JSR CHRGET
380         CMP #PLUSZ
390         BEQ RUTIN
400         JSR RINPUT
410         JMP INTER
420 RUTIN   JSR $B79B
430         CPX #$19
440         BCC FOLYT
450 HIBA    JMP $B248
460 FOLYT   STX $FB
470         JSR $E200
480         CPX #$28
490         BCS HIBA
500         TXA
510         TAY
520         LDX $FB
530         JSR $E50A
540         JSR CHRGET
550         JSR RINPUT
560         JMP INTER
570         .END

ZEILEN:48   SYMBOLE:13   FEHLER:0

BASVEC=0308  CHRGET=0073  CHRGOT=0079
FOLYT =C032  HIBA  =C02F  INPUT =0085
INTER =A7AE  PLUSZ =00AA  RINPUT=ABBF
RINTER=A7E7  RUTIN =C028  TALAL =C01B
VIZSG =C00E
```

1. lista

Egy új BASIC utasítás

Ez a gépi szubrutin az adatbevitelt könnyíti meg a képernyőszervezésnél. A C64-ből hiányzik az INPUT utasításnak az a fajta megoldása, hogy a képernyő tetszőleges sorába és oszlopába vihessük be az adatot. Ez csak a fénypontvezérlő karakterekkel lehetséges, vagy pedig bonyolult módon összePOKE-olva a gépet. Ezen a problémán segít a program, amely az INPUT+ utasítás használatát teszi lehetővé. *(1. lista)*

Az utasítás szintaxisa:

INPUT + sor, oszlop, "szöveg"; A

ahol a szöveg jelentése ugyanaz, mint az eredeti INPUT utasításnál, és "A" lehet sztring vagy numerikus adat. Az utasítás aktivizálása a program beírása után: SYS 49152. Ezek után az utasítás mindaddig érvényben marad, amíg a gépet ki nem kapcsoljuk.

Természetesen az INPUT+ mellett a régi INPUT is használható. Azok számára, akiknek nincs monitoruk, közlöm a *2. listát*, a BASIC betöltőprogramot.

KEINDL LÁSZLÓ

```
10 FORI=49152TO49226
20 READJ:S=S+J
30 POKEI,J
40 NEXTI
50 IFS<>8937THEN65
60 SYS49152:NEW
65 PRINT"HIBA AZ ADATOKBAN!":END
70 DATA 169, 14,160,192,141,  8,  3
71 DATA 140,  9,  3, 96,234,234,234
72 DATA  32,115,  0,201,133,240,  6
73 DATA  32,121,  0, 76,231,167, 32
74 DATA 115,  0,201,170,240,  6, 32
75 DATA 191,171, 76,174,167, 32,155
76 DATA 183,224, 25,144,  3, 76, 72
77 DATA 178,134,251, 32,  0,226,224
78 DATA  40,176,244,138,168,166,251
79 DATA  32, 10,229, 32,115,  0, 32
80 DATA 191,171, 76,174,167
```

2. lista

GET# helyett INPUT#

Ismeretes, hogy a Commodore 64 INPUT# utasítása maximálisan 88 (más forrás szerint 80) karaktert tud egyszerre beolvasni. Ennél hosszabb rekord beolvasására a szakirodalom a GET# utasítás vagy gépi kódú szubrutin használatát javasolja.

A GET# utasítással történő beolvasás legnagyobb hátránya, hogy rendkívül időigényes, és így az adatfeldolgozást jelentős mértékben lelassítja. Ennek kiküszöbölésére egyfajta olyan, BASIC nyelven írt programrészt mutatunk be, amely 88 karakternél hosszabb, példánkban a maximális 254 (ebből „hasznos" 251) karakterből álló rekordot INPUT# utasítással olvas be. A példát egy, a legkényelmesebben alkalmazható relatív fájlra alakítottuk ki.

Az egyszerűség kedvéért változóként egy 10 elemű, 251 karakterből, 0—9-ig terjedő számokból álló tömböt alkalmaztunk (64 $ (I)).

A programrész működésének lényege, hogy az egyes tömbelemeket három változóra (V1$, V2$, V3$) bontjuk, ezeket rögzítjük, majd az adatokat ugyanilyen formában olvassuk vissza. A rögzítéskor és a beolvasáskor a relatív fájl rekordjain belül a pointert szintén azonos értékekre állítjuk be. Megjegyezzük, hogy már az adatbevitelkor célszerű a ":" és a ";"-t letiltani, mert ezeket a karaktereket a gép utasításként értelmezi. Ennek megfelelően ajánlatos a mezőket más jellel elválasztani.

A program 45Ø-es sorában található IF utasítás azt a célt szolgálja, hogy üres sztring esetén a gép ne adjon ki STRING TOO LONG hibajelzést, ezért már a bevitelnél minden egyes rekordba egy "0" jelet tettünk be.

Ugyanazt a rekordot GET#, illetve INPUT# utasítással beolvasva, az első esetben a művelet 14, a másodikban 0,6 másodpercig tart; ez már jelentős mértékű megtakarítás. Megjegyezzük, hogy célszerű új

vagy újraformázott adatlemezzel dolgozni, mert a lemez és a meghajtó kapacitásának végső határán vagyunk. Máskülönben STRING TOO LONG hibajelzést kapunk.

A felhasználás során természetesen a változókon belül tetszőleges hosszúságú mezők alakíthatók ki. 251-nél rövidebb rekord esetén pedig annak csak első és/vagy második részét kell beolvasni, illetve rögzíteni. Sőt a példában alkalmazott egyetlen változó (GY $ (I)) helyett lehet több változót is használni. Ilyenkor csak arra kell vigyázni, hogy rögzítéskor a változókat CHR$ (13)-mal válasszuk el, még a pointer újrabeállítása előtt is. Például:

```
300 P=1: GOSUB 600: REM REKORD
    ÉS POINTER ÁLLÍTÁS
310 PRINT# 5, V 1 $; CHR$ (13); . . .; V
    4 $; CHR$ (13);
320 GOSUB 540: P=88: GOSUB 600:
    REM REKORD ÉS POINTER ÁLLÍ-
    TÁS
330 PRINT# 5, V 5 $; CHR$ (13); . . .; V
    9 $
340 stb. . . .
```

DR. KIRILLY ANDRÁS

```
10 REM **********************
20 REM * GY$(I) LETREHOZASA *
30 REM **********************
40 PRINTCHR$(147)
50 DIMGY$(11)
60 FORI=1TO10:FORJ=1TO25:FORK=0TO9
70 GY$(I)=GY$(I)+MID$(STR$(K),2,1)
80 NEXTK:NEXTJ:NEXTI
90 REM  ************************************
100 REM * ADATALLOMANY HELYENEK LEFOGLALASA *
110 REM ************************************
120 OPEN15,8,15,"I0"
130 OPEN5,8,9,"PROBA,L,"+CHR$(254)
140 GOSUB550
150 I=11
160 P=1:GOSUB610
170 PRINT#5,"o":GOSUB550
180 FORI=1TO10:P=1:GOSUB610
190 PRINT#5,"o":GOSUB550:PRINT"■■■■■■■■■■■■■■■■■■■■MEGNYITAS!"I
200 NEXTI:CLOSE5:CLOSE15
210 REM ******************
220 REM * ADATROGZITES *
230 REM ******************
240 OPEN15,8,15,"I0":OPEN5,8,9,"PROBA"
250 PRINTCHR$(147)
260 FORI=1TO10:GY$(I)=GY$(I)+"A":REM * KIEGESZITES 251 KARAKTERRE *
270 REM * GY$(I) SZETBONTASA *
280 V1$=LEFT$(GY$(I),84):V2$=MID$(GY$(I),85,84)
290 V3$=RIGHT$(GY$(I),83)
300 P=1:GOSUB610:REM * P 1.BEALLITASA *
310 PRINT#5,V1$CHR$(13);:GOSUB550
320 P=86:GOSUB640:REM * P 2.BEALLITASA *
330 PRINT#5,V2$CHR$(13);:GOSUB550
340 P=171:GOSUB640:REM * P 3.BEALLITASA *
350 PRINT#5,V3$:GOSUB550
360 NEXTI
370 CLOSE5:CLOSE15
380 REM ******************
390 REM * ADATBEOLVASAS *
400 REM ******************
410 CLR:DIMGY$(11)
420 OPEN15,8,15,"I0":OPEN5,8,9,"PROBA"
430 FORI=1TO10
440 PRINT"□   **I"**■■"
450 P=1:GOSUB610:GET#5,V1$:IFV1$="0"THEN530:REM * URES REKORD FIGYELESE *
460 V1$="":P=1:GOSUB610:INPUT#5,V1$:REM * POINTER 1.BEALLITASA *
470 GOSUB550:P=86:GOSUB640:INPUT#5,V2$:REM * POINTER 2.BEALLITASA *
480 GOSUB550:P=171:GOSUB640:INPUT#5,V3$:REM * POINTER 3.BEALLITASA *
490 GY$(I)=V1$+V2$+V3$
500 PRINTGY$(I)
510 V1$="":V2$="":V3$=""
520 FORJ=1TO1000:NEXTJ:REM * SZUNET *
530 NEXTI:CLOSE5:CLOSE15:END
540 REM ******************
550 REM * HIBA CSATORNA *
560 REM ******************
570 INPUT#15,A,B$,C,D
580 IFAC20RA=50THEN RETURN
590 PRINTA;B$;C,D:STOP:RETURN
600 REM ********************************
610 REM * REKORD ES POINTER ALLITASA *
620 REM ********************************
630 H=INT(I/256):L=I-H■256
640 PRINT#15,"P"+CHR$(9)+CHR$(L)+CHR$(H)+CHR$(P)
650 RETURN
```

COMMODORE 64

Beviteli rutin

```
0 REM * GEPI RUTIN BETOLTES *
1 IFPEEK(49153)=253THEN10
2 G=49151:FORI=1TO27:READX:POKEG+I,X:V=V+X
3 NEXTI:IFVC>3618THENPRINT"■■DATA HIBA!"
5 DATA32,253,174,32,158,183,138,72,32,253
6 DATA174,32,158,183,104,168,24,32,240,255
7 DATA32,253,174,32,164,179,96
9 REM * BEMUTATO PROGRAM *
10 PRINT"■■■":POKE53280,15:POKE53281,15
15 PRINT"■■■INPUT RUTIN BEMUTATO !"
16 PRINT"■■                    ■■"
20 INPUT"    HANYADIK SOR !";D
25 INPUT"    OSZLOP       !";C0
30 INPUT"    ■■■RAM, ■■■ETU     ";S$
31 IFS$<>"S"THEN35
32 INPUT"    TIZEDESEK    !";T
33 INPUT"    KITOLTO KAR. !";K$
35 INPUT"    ■■■EVEKEZ ■■■RAL !";R$
40 INPUT"    MEGNEVEZES   !";H$
45 INPUT"    MAX.KARAKTER !";MK
50 PRINT"□"
70 GOSUB60000
200 REM * FELDOLGOZO PROGRAM *
205 PRINT"■■■■■■■■■■■■■■■■";D$
210 IFS$="S"THEN250
220 PRINT"■■■■■■■ BEVITT SZOVEGES VALTOZO" :GOTO500
250 B=VAL(D$):N=B12:PRINT"■■■■■■■■";N
260 PRINT"■■■■■■■ SZAM NEOVZETE"
500 END
60000 REM * INPUT RUTIN *
60001 M=LEN(H$):C=C0+M+2
60002 IFD>20ORC0+M+2+MK>40ORMK=0THEN60600
60003 SYS40■1024,C0,D,H$;":";:POKE56325,25
60004 SYS40■1024,C,D,":";                    "
60005 POKE214,D:POKE211,C:SYS58640:POKE204,0
60010 GET#:IFA■=""THEN60010
60015 IFPEEK(207)>0THEN60015
60018 POKE204,1
60020 A=ASC(A$)
60025 IFA=13ANDS$="S"THENGOSUB60000
60030 IFA=13THEN60500
60040 IFA=92THENB■="":GOTO60001
60042 IFS$="S"RNDVAL(A$)■RNDRAC)46:
60043 RNDR#C)"0"RNDR#C)"-"THENB■="":GOTO60001
60044 IFC-(C0+M+2)>=0"THENGOTO60010
60045 IFR■="R"THENPOKE199,1
60050 SYS40■1024,C,D,A$
60055 POKE214,D:POKE211,C+1:SYS58640:POKE204,0
60060 C=C+1:B■=B■+R$
60070 GOTO60005
60500 IFPEEK(207)>THEN60500
60550 POKE204,1:FORI=1TO2000:NEXT
60580 POKE56325,60:RETURN
60600 PRINT"■HIBAS PARAMETEREZES!■■"
60610 IFD>20THENPRINT"SOR    ";D
60620 IFC0+M+2+MC>40THENPRINT"HEM FER EL A SORBAN!"
60630 IFMK=0THENPRINT"MAX KARAKTER = 0"
60640 PRINT"■■FOLLVTATJA? (1/N)"
60650 GET#: IFB■=""THEN60650
60660 IFB■="1"THENRETURN
60665 IFB■<>"N"THEN60650

60665 IFB■<>"N"THEN60650
60670 POKE56325,60:END
60680 IFK■="THENK■CHR■(32)
60805 TU=10↑T
60810 B=VAL(B$):B=(INT(TU■B+.5))/TU
60812 B=B■1/10↑(T■1)
60815 BU■=STR■(B):HU=LEN(BU■)
60818 BU■=LEFT■(BU■,HU-1)
60820 IFHU>MKTHEN60845
60830 BU■=■■BU■
60840 IFLEN(BU■)<MKTHEN60830
60845 IFR■="R"THENPOKE199,1
60850 SYS40■1024,C0+M+2,D,BU■
60860 FORI=1TO2000:NEXT:POKE199,0:RETURN
```

Az UNIN rutint olvasva (1985/6. szám), illetőleg olvashatatlansága, bonyolultsága és változópazarlása miatt kezdtem hozzá egy beviteli rutin megírásához. A program kurzorhasználatot, bevitelellenőrzést és formázott kiíratást is tartalmaz. Egy jól ismert gépi rutint használ, amely megtalálható Lángos István: A Commodore 64 mikrogép kezelése és programozása című könyvében, és aminek talán új oldalát sikerült megtalálni. A gépi rutin bevitelére a program elején DATA listát adok; haladóbbak az assembly lista alapján készíthetnek maguknak programfájlt. Talán meglepő, hogy a nem megfelelő bevitel töröl; ez a biztonságra törekvésnek köszönhető. Az esetleg elrontott billentyűzést a "font" billentyűvel lehet javítani.

Természetesen a program továbbfejleszthető, ezért is adom közre. Például az elkészült oldalak ellenőrizhetők a rutin kismértékű módosításával: a kurzor ugorjon mindig a következő adatra.

A használt numerikus változók: A,B,C,C0,D, HU,M,MK,T,TU,(X,Y), a szöveges változók: A$,B$,BU$,K$,M$,R$,S$.

Felhívom a figyelmet arra, hogy a 60042 és a 60043 sor egy sort alkot, csak nyomdatechnikai okokból vannak különválasztva.

MOCSÁRI ATTILA

UNIN példaprogram

A mintapélda remélhetőleg bebizonyítja, hogy milyen hatékony, gyors programozást tesz lehetővé a rutin használata (1985/6. szám). A példa ugyan töredékét mutatja be a rutin adta lehetőségeknek, de így is jól érzékelteti a paraméterezhetőség előnyeit.

A minta max. 10 ember adatait képes felvenni. Egy ember adatait egy képernyőn vihetjük fel vagy módosíthatjuk, s lapozással juthatunk egy másik egyén kartonjára. Bármelyik kartonon bármit módosíthatunk. A hibaüzenet kezelésére is láthatunk példát.

```
10 REM**********************************
20 REM*****                        *****
30 REM***** U N I V E R Z A L I S *****
40 REM*****       I N P U T        *****
50 REM*****       P E L D A        *****
60 REM*****                        *****
70 REM***** KESZITETTE : BORBELY   *****
80 REM*****              JOZSEF    *****
85 REM*****                        *****
90 REM**********************************
99 GOSUB100:GOTO1000
100 REM-------------------------------
110 REM-----                     -----
120 REM----- 1 0 0 - 5 4 5 - SOROK -----
130 REM-----                     -----
140 REM-----    I N P U T        -----
160 REM-----    R U T I N        -----
170 REM-----                     -----
545 REM-------------------------------
1000 P$(1)="00021220":P$(2)="09041611"
1005 P$(3)="00061620":P$(4)="07081805"
1010 P$(5)="00101620":P$(6)="00121615"
1015 P$(7)="48142606"
1020 PRINT"■":TAB(13)"PELDA PROGRAM■"
1025 PRINT" 1. NEV    :■"
1027 PRINT" 2. SZEM.SZAM :■"
1030 PRINT" 3. SZUL.HELY :■"
1035 PRINT" 4. SULYVALTOZAS :          KG■"
1037 PRINT" 5. MUNKAHELY :■"
1040 PRINT" 6. BEOSZTASA :■"
1045 PRINT" 7. UTOLSO MERES DATUMA :■■"
1050 PRINTTAB(6)"MODOSIT?";
1052 PRINTTAB(20)"MELYIK SOR?■"
1055 PRINTTAB(8)"LAPOZ?      + - N■"
1057 PRINTTAB(9)"VEGE?";
1060 CO=7:B6=7:B8=0:B0=0:B7=0
1065 FORJ=1TO7:GOSUB1180:NEXT
1070 L=17:C=15:B1=0:GOSUB535:J$="Y"
1075 GOSUB1200:IFA$<>"Y"THEN1100
1080 L=17:C=33:B1=9:J$="1":GOSUB1200:J=A
1085 IFA<10RA>7THENGOSUB1210:GOTO1080
1090 J$=TA$(LAP,J):GOSUB540:A=1
1095 GOSUB1180:GOTO1070
1100 L=19:C=15:J$="+":GOSUB1200
1110 IFA$="+"ANDLAP<9THENLAP=LAP+1
1120 IFA$="-"ANDLAP>0THENLAP=LAP-1
1130 IFA$="N"THEN1160
1135 IFA$<>"+"ANDA$<>"-"THEN1100
1140 IFTA$(LAP,1)=""THEN1020
1150 PRINT"■■":FORJ=1TO7
1155 PRINTTAB(VAL(MID$(P$(J),5,2))),
1157 PRINTTA$(LAP,J)"■":NEXT:GOTO1100
1160 L=21:J$="N":GOSUB1200
1165 IFA$<>"I"THEN1070
1170 END
1180 B1=VAL(LEFT$(P$(J),2))
1185 L=VAL(MID$(P$(J),3,2))
1187 C=VAL(MID$(P$(J),5,2))
1190 B=VAL(RIGHT$(P$(J),2)):GOSUB100
1195 TA$(LAP,J)=A$:RETURN
1200 B7=7:A=1:GOSUB540:GOTO100
1210 L=24:C=0:B7=0:B=0:GOSUB530
1215 PRINT" HIBAS VALASZ !";:GOSUB100
1217 PRINT"                ";:RETURN
1230 REM-------------------------------
1300 REM ALTALABAN : X$ MODOSITASA :
1310 REM
1320 REM B7=7 : GOSUB 535 : J$=X$
1325 REM GOSUB 545 : GOSUB 100 : X$=A$
```

A programsorok magyarázata:

99	Kezdeti feltételek beállítása, ugrás a főprogramra.
1000—1015	A bemeneti mezők egyedi paraméterei P$ ( ) sztring tömbként vannak megadva, egy tömbelem egy mezőt határoz meg. Egy-egy tömbelemben 2 karakterenként egy paraméter értéke helyezkedik el. Így egy mezőhöz összesen 4-fajta egyéni tulajdonság kapcsolódhat, sorrendben: a mező típusa, képernyősor, oszlop, mezőhossz.
1020—1057	Egy karton állandó részeinek kiírása a képernyőre.
1060	Állandó paraméterek értékadása.
1065	Új karton mezőinek kitöltése.
1070	Módosítás? Ha nem, ugrás 1100-ra.
1080	A módosítandó sor sorszámának bekérése és ellenőrzése.
1090	A megadott mező módosítása. A módosítandó sztring J$-on keresztül kerül A$-ba az 535-ös és 540-es rutinok segítségével!. Ugrás 1070-re.
1100—1135	Lapoz előre (+), hátra (−) vagy ugrás 1190-re (N).
1140	Üres lap esetén új lap kitöltése: ugrás 1020-ra.
1150	Új lap kiírása, ugrás 1100-ra.
1160	Vége? Ha nem, ugrás 1070-re.
1170	Vége.
1180—1195	A módosítandó mező paramétereinek beállítása, módosítás.
1210	Hibakiíró szubrutin.

BORBÉLY JÓZSEF

ADOK – VESZEK – CSERÉLEK

Ebben a rovatunkban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás: közleteknek gépelt soronként (60 karakter) 100 Ft, magánszemélyeknek az első sor 50 Ft, minden további sor 20 Ft. Az NJSZT tagjainak az első három sor ingyenes. Hirdetéseiket a szerkesztőség címére várjuk.

■ TRS 80 MC—10 sz. géphez 16 k-s bővítő, interfész-kábelek, könyvek, programok és 1 db COMPUTEK botkormány eladó vagy elcserélhető. Képíró Róbert, Bp., Imre u. 5. 1093. Telefon: 378-075.

■ Sinclair Spectrum megkopott fedőlapját (fém) olcsón és gyorsan újjávarázsolom. Képíró Róbert, Bp., Imre u. 5. 1093. Telefon: 378-075.

■ Commodore 64 gépre eredeti OXFORD—PASCAL programnyelv dokumentációval eladó. Leskóné Puskás Judit, Bp., Aszú u. 25. 1028. Telefon: 830-443 vagy 298-750.

■ Commodore VC20 játék- és egyéb programokat cserélnék. Keresem a Sakk, Super Expander és bővítő leírást másolásra. Rába Kálmán, Szombathely, Bolyai János u. 13. F. IV. 19. 9700.

■ C64-hez hard disket, Winchestert vennék. Programírást és oktatást vállalok minden szinten (BASIC, FORTH). Telefon: 154-352.

■ A Mikroszámítógép Magazin első 13 száma — csak egyben — eladó. Ára 400 forint. Postán utánvéttel elküldöm. Kovács László, Karcag, Zádor u. 8. 5300.

■ VC20 számítógép magnóval, botkormánnyal, jelentős mennyiségű programmal és dokumentációval együtt 19 000 forintért eladó. Setény János, Békéssámson, Vöröskatonák u. 97. 5946.

Fejlesztési Intézetünkbe felvételre keresünk korszerű mikroprocesszoros terminálok és berendezések HARDWARE—SOFTWARE-fejlesztéséhez villamosmérnököket, készülék konstrukciós kialakításához felsőfokú végzettségű szakembereket.

RUGALMAS MUNKAIDŐ
KIEMELT FIZETÉS
MAGAS JÖVEDELEM

Jelentkezés:
TELEFONGYÁR
1143 Bp. Hungária krt. 126-132.
Tel.: 634-330.

TERTA

Barátunk, a számítógép

Vélt és valós tendenciák

(Mottó: A szabad értelem próbája nem az értékek tagadásában rejlik. Hanem abban, hogy bölcsen tud választani az értékek közül. — A. L. Sacker)

Ülök a televízió előtt. 12 éves, mosolygós, szemüveges fiúcska áll valamilyen szerkentyű mellett. A riporter biztató kérdése: — Ugye, Pistike, te már valóságos zsonglőr vagy a programozásban? Pistike határozottan bólint. A riporter tovább évődik: — És ugye kisfiam, ugye apukád, ugye nem kezeli a számítógépet? Mire a gyerek kihúzza magát, és jön a meglepő válasz: — Nem, apuka nem tudja kezelni a számítógépet.

Döbbenetes, hogy a józan hittel szemben milyen gyorsan terjednek a hamis mítoszok, amelyeknek egykettőre bedőlünk. Szomorú, hogy mennyire nem tudják tálalni a barátunk népszerűsítésére szolgáló műsorokat. Holott köztudott, hogy az újjal szembeni természetes ellenérzést csak igen nagy tapintattal lehet leküzdeni.

Emiatt a félrekezelés miatt ragadtam írógépet. Szeretnék hitet adni a felnőtteknek, barátjukká tenni a számítógépet. A fiatalokat pedig realitásra kívánnám inteni, mint amolyan középkorú. Végeredményben a vélt és valós tendenciákról lesz szó ebben a cikkben. Remélem, hogy sikerül majd tisztáznom egy alapkérdést: az eszközök szerepét életünkben. Nem fogok lapocskákról, programokról, billentyűkről beszélni; majd eljön annak is az ideje. Ha már látjuk a lényeget, amit nem takarnak el a részletek.

A felnőttekhez

Apa a hosszú tárgyalások után fáradtan ül a tévé előtt. Ha most jó krimi kezdődne, akkor apa (lelkiismerete) kiszólna a konyhába anyához: — Fiam, hagyd azt most! Jó a műsor!

De most a tévében éppen arról van szó, hogy a bal felső sarokba írjunk egy furcsa angol szót. Felkiáltójel, pontosvessző, F6 billentyű . . . Apa dörmög valamit, és átkapcsol a focimeccsre.

Apa vállalati vezető. Naponta dönt emberekről, gépekről, milliókról. Aranygyűrűs diplomás, több találmány szerzője. Nem az a legfontosabb ismérve, hogy nem tud számítógépet kezelni.

Más családban anya keze alatt egyszerre 15-20 hatalmas szövőgépsor zakatol. A második műszak után kicsit fáradt ahhoz, hogy számítógépül tanuljon. Ahhoz különben is matematika kell, ami anyunak — a kosztpénzt kivéve — nem erős oldala.

Tájékoztatásunk félig átgondolt. Jószándékú, de ügyetlen. Nem épít a rétegkommunikáció titkaira, amiket még középkorú szomszédnőnk is átérez. Így a kommunikáció az amúgy is értőknek szól, és nem ér el másokhoz, hogy bővítse barátunk táborát.

Kedves apukák és anyukák! Apa igen kis erőfeszítéssel saját maga kérhetné le az aktuális trendet, a fedezetszámítást, az árkalkulációt. Úgy, hogy saját maga kezelné a gépet, de nem furcsán írt vezényszavakkal. Anya pedig mindig csodálatos érzékkel rendelkezett a színek-minták iránt. Géppel tervezhetne csodálatos szöveteket, amire mindig vágyott. Nem kellene sóhajtva — de büszkén — legyintenie: majd a kisfiam . . .!

Higgyék el a kedves felnőttek, hogy barátunk kezelése nem nagyobb ügy — kellő felkészítés esetén! —, mint mondjuk az autóvezetés. Remélem, hogy e cikksorozat végére ezt így fogják látni. Valóban barátunk a számítógép!

A fiatalokhoz

Kedves, ifjú szomszédom komoly számítógépet kapott. Nem játékot, amin két cowboy üldöz három bölényt, miközben őket viperák kergetik. Édesanyja ellenzi a durvaságot terjesztő „játékokat". A kis szomszéd lelkesen készít programokat BASIC nyelven, és szentül hisz a kiadott jelszóban: a számítógép fejleszti a fiatalok feladatmegoldási képességét és gondolkodási készségét.

Nem dicsérlek szembe, csak hátad mögött mondom: az igazság pont fordított. Mert van akaraterőd, van feladatmegoldási készséged és képességed, még ma is kitartasz újabb barátod mellett. Nem úgy, mint Feri, akinél a gép státusszimbólum, és aki az előregyártott program ötödik futása után unatkozott. Apja nem vett újabbat, mert nem is olyan olcsó. Ő meg nem írt programot, mert az fáradságos. A gép porosodik: nem lett barát.

Ifjú szomszédom! Örülök sikereidnek, de tanulj meg egyet: nem jelenti képességeid csúcsát, hogy tudsz BASIC-ben programozni. A BASIC nem a jövő század kommunikációs nyelve, mint számos helyen hangoztatják. Kicsit nehézkes, kicsit unalmas. Szóval: van más is. Remélem, hogy a cikksorozat végére belátod ezt. Addig is megegyezhetünk abban, hogy még e században is irdatlan sokat kell tanulnod ahhoz, hogy kis barátodból, a számítógépből kihozzad, ami benne van.

Gépekről, komolyan

Számítógép és számítógép között akkora különbség van, mint jármű és jármű között. Nem megyek a szomszéd közértbe autóval, nem biciklizem Párizsig, és az óceán fölött repülni fogok. Be kellene már végre látni, hogy egy-egy szervezet komoly információs feladataira csak meghatározott képességű számítógépek adhatnak megoldást.

Ne csináljunk erényt a kényszerből! Ha nem telik repülőre, akkor autóval nem fogok áthatolni Amerikába! A személyi számítógépek adott célokra valók. Igaz, e fránya kis masinák egyre többet és többet tudnak. Már nem csak „személyiek". De még nem pótolják a nagybácsit, az „igazi" számítógépet.

Utóbbiak kezelése nem mindennapos feladat, hanem hivatás. Pistike, aki ma kitűnően kezeli a számítógépet, holnap már alkalmas lesz arra a feladatra is, ha kitart. Örülnünk kell tehát minden tévés és egyéb oktatásnak, amiben a mi Pistink részesül. Bár a felnőttek is vennék olyan komolyan barátunkat, mint Pistike teszi!

Nem úgy, mint egyes hozzászólók. Felnőttek, akik semmit sem értenek. Idézek:

„Vettünk 12 Commodore 64-es gépet. Ezzel nálunk a elektronizáció befejeződött." És az élet? Azóta a nyilatkozó karrierje fejeződött be. Mert a gép használatához rendszerekre, programokra van szükség — és sok-sok pénzre.

„A készletrendszert számítógépre ültetjük." Annál a nagyvállalatnál annyiféle tétel mellett aligha lesz elegendő az a kis IBM PC.

A kulcskérdés

Van-e szüksége egy ekkorka országnak arra, hogy tízezrek tudjanak BASIC-ben programozni? Nem kívánok itt ellenpropagandát kifejteni. Csak a felhangokat akarom letörni. Apa, az ő fizetésével csak ne programozzon! Anya pedig ne tanuljon matematikát, ha már eddig nem tette!

A személyi számítógépek forgalmazásának hatalmas lendülete az utóbbi két évben megtört. Elsősorban azért, mert egyre többen ismerték fel, hogy ez a gép nem olcsó játékszer. A hivatástudó fiatalok százai maradnak meg hűségesen a barát mellett, de nem mindenkiből lesz programozó, aki számítógépet látott-kezelt. Mi tehát a helyes út?

Közeledik anya születésnapja. Pisti újdonsággal akarja őt meglepni. Bár már érmet is nyert programozási versenyen, most valami nem megy, nem megy . . . És Pisti ekkor apuhoz megy és kérdez. — Beszélj magyarul, fiam! — inti apja. Nem utasít el, nem mondja: neked vettem. Apa figyel és magyaráz. És a magyarázatok végére Pistikéből István lesz: megérti, hogy nem elágazás, ciklus, RUN és CASE a lényeg. A feladat a kemény dió, amit apu játszva feltört, pedig nem is tudja kezelni a számítógépet. Pistike megtanulta, hogy az új seprő jól seper, de a régi tudja, hogy hol van a piszok. Pistiből most már bizonyosan jó szakember lesz.

Anya a konyhában matat: hová is tettem a bejgli receptjét? Pisti büszkén mondja: kidobtam. Nem, ne haragudj. Itt van, csak kezeld a számítógépet! — No de fiam, ne bolondozz már! — néz vissza aggódva anya a konyha felé. — Anya, csak ezt a két gombot nyomd le! Úgy! Most a kérdőjel után írd azt, hogy bejgli! — utasítja Pisti. Anya vonakodva megteszi, amit okos fia mond. Hátrahőköl, amikor felpereg a nyomtató, és duruzsoló hangon kezdi: A mákosbejgli receptje. Végy . . .

Szép volt a születésnapi ajándék. Emberként is és programozóként is érettté vált Pisti. Olyanná, aki képes felfogni mások valódi információs igényét. Aki barátot akar szerezni az ő barátjának azzal, hogy nem kényszeríti szokatlan trükkökre. Hanem úgy készíti fel a találkozást, hogy az mindkét félben, de főleg az emberben kedves és maradandó — hasznos? — emlék legyen. — Lám, anya matematika nélkül is tudja kezelni a gépet! — nézett össze apa és fia.

Az egész ilyen egyszerű. Kell, aki feladatot lát. Kell, aki megoldást kínál. Már rutin a megvalósítás. Kell sok boldog, gépbarát anyuka, és szükség van az igényt megvalósító tehetséges ifjakra. Kezdetnek csak ennyit akartam mondani.

DR. HALASSY BÉLA

Az Építsünk számítógépet! sorozat egy hibájáról

Az 1985/1. és 2. számban közölt videoNYÁK rajzában „hibákat" találtam, ezekkel foglalkozom.

Az EPROM (IC 41) kiválasztása a PIA (IC 33) PBØ adatvonalával történik. Ha az IC 23A 7. lábán „1" szint van, akkor az EPROM van kiválasztva. Nos, ez így nem igaz, mert az IC 23A 7. lába le van kötve a negatívra. Ugyanakkor ez a vezeték egy, de ha jobban megfigyeljük, két 3K ellenállással a pozitívra van kötve. Miért?

Az IC 41 alatt található a grafika kiválasztása. Nem értem, hogy mi az, ami az IC 30, 29-es bemeneteire kapcsolódik?

Az IC 46 3. lába (bemenet) az IC 17 9. lábára (bemenet) van kötve. Miért? Ugyanakkor az IC 17 kimenetei nincsenek felhasználva!

Számomra még mindig érthetetlen, hogy az IC 46-tal miért végezteti el még egyszer azokat a feladatokat, amiket az IC 22 már elvégez, és ezen IC-k 7, 12, 9-es lábait összeköti. Miért?

A video RAM-ok címzésénél két eset lehetséges: 1. a 6845 címezi a RAM-ot; 2. a processzor címezi. Szerintem a 6845 nem képes az 1 k tárterületeket kiválasztani, mivel az IC 20 bemeneteire közvetlenül a processzor AØ, A1 címvonala kerül. Ezzel szemben a leírásban VA1Ø, 11-es vonalak szerepelnek. Így a 6845 már képes a RAM-csoportok kiválasztására, mivel az IC 26 bemenetei pozitívon vannak. Ebből következik, hogy az M2-es vezeték logikai „Ø" állapotban van, így az MX-K IC 21, 22, 23 a 6845 címvonalait kapcsolják a memóriákra. Itt megint felmerül egy probléma. Most a processzor nem képes a RAM-ok címzésére! Ennek ui. az a feltétele, hogy az M2-es vonal „1" szinten legyen, ez pedig a jelenlegi kapcsolásban nem jöhet létre tartósan. Az IC 26 valamelyik bemenetének „Ø" állapotba kell kerülnie. Ez egy pillanatra meg is történik, amikor a 6845 A10, 11, 12 címvonalai „1" állapotban vannak. Ekkor az M2-es vonal „1" szintű. Az IC 46 3-as bemenetét az IC 20 3-as bemenetére kapcsolja. Amennyiben az IC 46 3-as bemenete „Ø", akkor az M2-es vonal ismét „Ø" szintű, azaz a 6845 címezi a memóriát. Talán ez a pillanatnyi állapot elegendő arra, hogy a processzor a megfelelő memóriarekeszt átírja? Ha igen, az IC 46 3-as lábát a címbusz A12-es vezetékére kell kötni, hogy a processzor is a teljes tárterülettel tudjon dolgozni!

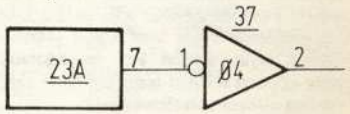
Kérem, amennyiben a hibák igaznak bizonyulnak, közöljék velem. Ha lehet, a Magazinban. Javasolnám a komplett kapcsolási rajz közlését.

CSÁNYI CSABA

A szerkesztő válaszol

A videokártya egy nem grafikus kártya módosításával készült. A módosítás miatt átkötések, egymásra építések és visszamaradt — már felesleges – vezetékek is találhatók. A lap 1985/2. számában például a 2. ábrán D7 az IC 29 14. bemenetén kívül egy szám nélküli volt alkatrész 20. lábára — pontosabban a foglalatra — is rá van csatlakoztatva. Ez után az általános megjegyzés után konkrétan válaszolok a kérdésekre.

1. EPROM-kiválasztás. Az 1985/2. számban a 2. ábrának hiányzik az IC 23A 7. láb után egy részlete:



2. Grafikakiválasztás. Erre az általános megjegyzés a magyarázat.

3. Memóriacímzés. Az 1985/1. szám 4. ábrájáról lemaradt: az IC 46 3. lábára és az IC 17 9. lábára (bemenetek) az A12 jel kerül. Az általános megjegyzés itt is érvényes.

4. Az IC 46 feladata. A módosítás többlet-RAM beépítését és ennek címzését jelentette.

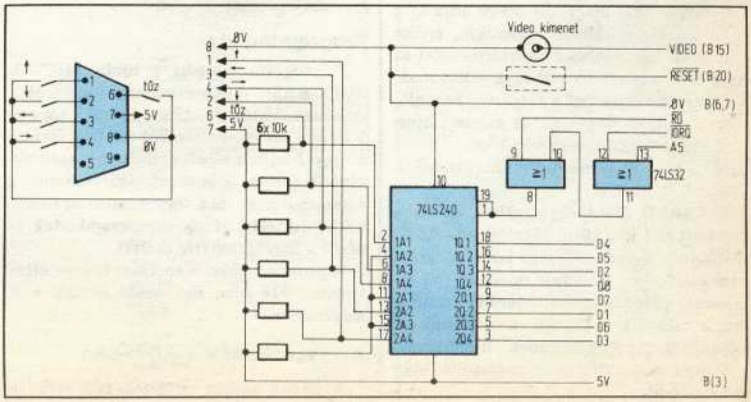
5. A video-RAM sávkiválasztása. Az 1985/1. szám 4. ábrájáról lemaradt: az IC 46 4. lábára és az IC 20 3. lábára (bemenetek) a VA12 jel kerül. Nem a rajz, hanem a szöveg a helyes. Így az IC 20 1. és 2. lábaira (bemenet) nem AØ és A1, hanem VA1Ø és VA11 kerül.

6. A teljes rajz közlésére jelenleg nincs mód.

Lehet egyszerűbben is

Az 1986. március havi szám 28. oldalán megjelent ZX-Spectrum Kempstone botkormányillesztő kapcsolási rajzához szeretnék kiegészítést fűzni. Kétségtelen, hogy az áramkör működik, de szükségtelenül bonyolult. Én egy olyat mutatok be, amely csak két IC-t (74LS240 és 74LS332) tartalmaz, szemben a három IC-t tartalmazó eredetivel, viszont ugyanazokat a funkciókat látja el.

PINTÉR GÁBOR
Kiskunhalas



PTA-4000 BASIC

Sorozatunkkal egyidőben a Rádiótechnika c. lap bemutatja a gép alap BASIC-jét. Itt ennek csak néhány, a gépi kódú programozásnál használt utasítását ismertetem.

Néhány utasítás

Az utasítások formátuma és jelentése a következő.

NEW kifejezés

Ezt a parancsot, mely programban is használható, a használati utasítás is ismerteti, ha a kifejezés egy állandó, éspedig Ø. Más érték is adható azonban. Ebben az esetben a kifejezés értéke adja meg a BASIC tárolóterületének kezdőcímét. Természetesen olyan értéket kell megadnunk, amely egyrészt a RAM-területen belül van, másrészt a foglalt területen kívül.

STATUS kifejezés

A kifejezés értéke itt is korlátozott. Lehetséges értékei: Ø,1,2,3,4. A Ø érték esetén a szabad memória és az adatterület összegét adja vissza. Az 1 érték megadja a BASIC programterület nagyságát. A 2 a BASIC programterület végcíme utáni első címet, a 3 az adatterület kezdőcímét jelzi.

PEEK✻ kifejezés

Hasonlít az ismert függvényhez, de az ME1 memóriaterületen levő cím tartalmát szolgáltatja.

POKE 1. kifejezés, 2. kifejezés, 3. kifejezés . . .

Az első kifejezés a memóriacímet adja, a következő az oda tárolandó adatot, az ez után következő a következő memóriacímre tárolandót stb.

CALL kifejezés, változó

Változó és az azt megelőző vessző nélkül a kifejezés által megadott címre adja át a vezérlést. Ha a változó numerikus, értéke átkerül az Xreg-be, és a visszatérés előtt az Xreg értéke átadódik ennek a változónak. Karakteres változónál a folyamat hasonló, csak a változó nagysága az akkumulátorban, kezdőcíme az Xreg-ben lesz.

CSAVE M—1 „állománynév"; 1. kifejezés, 2. kifejezés, 3. kifejezés

A CSAVE M—1 egymagában átadja a vezérlést az 1 jelű külső készüléknek. Az állománynév a védendő gépi kódú program nem kötelezően kiadandó neve; az első kifejezés a védendő memóriaterület kezdőcíme, a második a végcím, a harmadik az ugyancsak nem kötelezően használható programindítási cím, mely visszatöltés után aktivizálódik.

CLOAD M—1 „állománynév"; kifejezés

A CLOAD M magát a beépített kazettásegység-illesztőt használva tölt be egy programot, mégpedig a szalagon levő következő gépi kódút; az állománynév használata esetén pedig a megnevezettet. A—1 használata ugyanazt jelenti, mint a CSAVE esetén. A kifejezés azt a címet adja meg, ahová a program töltődni kezd. Ha a CSAVE utasításnál használtuk a harmadik kifejezést, akkor a program automatikusan indul.

Kulcsszavak

A gép BASIC fordítóprogramját a MICROSOFT cég készítette, így a BASIC által használt szavak, az ún. kulcsszavak felismerése egy kódtáblázat tartalmával való összehasonlítással történik. Eltérően a legtöbb MICROSOFT-BASIC-től, itt minden kód 2 bájt hosszúságú. A használt kulcsszavakat és kódjukat az *1. táblázat* foglalja össze.

Adattárolás

Az adatokat 8-8 bájton tárolja a fordító, függetlenül az adat típusától. Mégis megkülönböztet háromféle adatot. Ezek közül csak a decimális szám tárolásánál használja ki a fenntartott 8 bájtot.

Decimális számok tárolásánál az első bájton a kitevő, a következőn az előjel, a következő 5 bájton a mantissza, az utolsón mindig ØØH található. Néhány változatot bemutatok a *2. táblázatban.* Ugyanitt található példa a bináris szám tárolására is. Ezeknél az első négy bájtot a fordító nem veszi figyelembe, a következő mindig B2H, az ezután következő két bájton található a binárisan tárolt szám, az utolsót ugyancsak nem értékeli a fordító.

Karaktersorozat tárolásánál *(3. táblázat)* az első négy bájt szintén közömbös, a következő értéke mindig DØH, az ezután következő két bájt a karaktersorozat tárolásának kezdőcíme, az utolsó pedig a sorozat hossza, binárisan tárolva.

Programtárolás

A program tárolása a többi MICROSOFT-BASIC-hez hasonlóan a kulcsszókódok használatával történik. Minden sor egy 2 bájton tárolt sorszámból (bináris forma) és egy 1 bájton tárolt sorhosszból (szintén bináris forma, a sorhossz számításánál a sorszámot nem, de a sorvégkaraktert beszámítja) áll, ezt követik a programkódok és végül a sorvégkarakter (ØDH).

A program végét a sorszám helyén elhelyezett FFH jelzi. Egy példa látható a *4. táblázatban.*

Kulcsszókódok a ROM-ban

A ROM három területén találhatók a kulcsszókódok: BØØØH-BØE9H, B8ØØH-B887H, C054H-C34EH.

Az *5. táblázat* az 1. táblázat kulcsszavait tartalmazza az ezen a ROM-területen belüli címükkel együtt. Néhány helyen a cím hiányzik. Ezek ott nem találhatók, de nem találhatók a ROM más részein sem. Ezt a táblázatot azért állítottam össze, hogy megkönnyítsem a ROM-ban keresést azoknak, akik nem kívánják kivárni a sorozat végét.

Üzenetek

A rendszer „bejelentkezésénél" használt üzenetek a következő címterületen találhatók *(6. táblázat):* C34FH-C36AH.

DR. SIMONYI ENDRE

1. táblázat. Kulcsszókódok

ABS	F170H	DTE	E884H
ACS	F174H	END	F18EH
AND	F150H	ERL	FØ53H
AREAD	F180H	ERN	FØ52H
ARUN	F181H	ERROR	F1B4H
ASC	F160H	EXP	F178H
ASN	F173H	FEED	FØBØH
ATN	F175H	FOR	F1A5H
BEEP	F182H	GCURSOR	FØ93H
BREAK	FØB3H	GLCURSOR	E682H
CALL	F18AH	GOSUB	F194H
CHAIN	FØB2H	GOTO	F192H
CHR$	F163H	GPRINT	FØ9FH
CLEAR	F187H	GRAD	F186H
CLOAD	FØ89H	GRAPH	E681H
CLS	FØ88H	IF	F196H
COM$	E858H	INKEY$	F15CH
CONSOLE	FØB1H	INPUT	FØ91H
CONT	F183H	INSTAT	E859H
COLOR	FØB5H	INT	F171H
COS	F17EH	LCURSOR	E683H
CSAVE	FØ95H	LEFT$	F17AH
CSIZE	E680H	LEN	F164H
CURSOR	FØ84H	LET	F198H
DATA	F18DH	LF	FØB6H
DEG	F165H	LINE	FØB7H
DEGREE	F18CH	LIST	FØ90H
DEV$	E857H	LLIST	FØB8H
DIM	F18BH	LN	F176H
DMS	F166H	LOCK	F1B5H
		LOG	F177H
LPRINT	FØB9H	ROTATE	E685H
MEM	F185H	RUN	F1A4H
MERGE	FØ8FH	SETCOM	E882H
MIND$	F17BH	SETDEV	E886H
NEW	F19BH	SGN	F179H
NEXT	F19AH	SIN	F17DH
NOT	F16DH	SORGN	E684H
OFF	F19EH	SPACE$	FØ61H
ON	F19CH	SQR	F16BH
OPN	F19DH	STATUS	F167H
OR	F151H	STEP	F1ADH
OUTSTAT	E880H	STOP	F1ACH
PAUSE	F1A2H	STR$	F161H
PEEK	F16FH	TAB	FØBBH
PEEK✻	F16EH	TAN	F17FH
PI	F15DH	TERMINAL	E883H
POINT	F168H	TEST	FØBCH

POKE	F1A1H	TEXT	E686H
POKE #	F1A0H	THEN	F1AEH
PRINT	F097H	TIME	F15BH
RADIAN	F1AAH	TO	F1B1H
RANDOM	F1A8H	TRANSMIT	E885H
READ	F1A6H	TROFF	F1B0H
REM	F1ABH	TRON	F1AFH
RESTORE	F1A7H	UNLOCK	F1B6H
RETURN	F199H	USING	F085H
RIGHT$	F172H	VAL	F162H
RINKEY$	E85AH	WAIT	F1B3H
RLINE	F0BAH	ZONE	F0B4H
RMT	E7A9H		
RND	F17CH		

2. táblázat. Számtárolás

Decimális
04H 00H 12H 34H 00H 00H 00H 00H = 1234
00H 00H 12H 34H 00H 00H 00H 00H = 1.234
FDH 00H 12H 34H 00H 00H 00H 00H = 0.001234
03H 80H 12H 34H 00H 00H 00H 00H = −1234
Bináris
XXH XXH XXH XXH B2H 7FH FFH XXH = 32767
XXH XXH XXH XXH B2H 80H 00H XXH = −32768

3. táblázat. Karaktersorozat-tárolás

XXH XXH XXH XXH D0H 7BH 22H 11H = A karaktersorozat a 7B22H címtől kezdve tárolódik, és 17 bájt hosszú

4. táblázat. Programtárolás

00H 14H	= 20
06H	= a sor karaktereinek száma (6, mert a sorvég is számít)
F0H 97H	= PRINT
34H	= "
42H	= B
34H	= "
0DH	= sor vége
00H 1EH	= 30
03H	= a sor karaktereinek száma
F1H ACH	= STOP
0DH	= sor vége
FFH	= program vége

5. táblázat. A kulcsszókódok címei

ABS	C067H	DTE	—
ACS	C07FH	END	C10DH
AND	C05FH	ERL	—
AREAD	C055H	ERN	—
ARUN	C08FH	ERROR	C11DH
ASC	C087H	EXP	C115H
ASN	C077H	FEED	—
ATN	C06FH	FOR	C127H
BEEP	C098H	GCURSOR	C14DH
BREAK	C35CH	GLCURSOR	B073H
CALL	C0D8H	GOSUB	C138H
CHAIN	B855H	GOTO	C12FH
CHR$	C0CFH	GPRINT	C142H
CLEAR	C0B5H	GRAD	C159H
CLOAD	B05FH	GRAPH	B069H
CLS	C0BFH	IF	C16CH
COM$	—	INKEY$	C17BH
CONSOLE	—	INPUT	C162H
CONT	C0A1H	INSTAT	—
COLOR	B055H	INT	C173H
COS	C0C7H	LCURSOR	B080H
CSAVE	B069H	LEFT$	C1AEH
CSIZE	B05FH	LEN	C1A6H
CURSOR	C0AAH	LET	C19EH
DATA	C104H	LF	B08CH
DEG	C0F4H	LINE	B093H
DEGREE	C0E9H	LIST	C186H
DEV$	—	LLIST	B09CH
DIM	C0D2H	LN	C197H
DMS	C0ECH	LOCK	C1B8H
		LOG	C18FH
LPRINT	B0A6H	ROTATE	B0BBH
MEM	C1C1H	RUN	C25EH
MERGE	B873H	SETCOM	—
MID$	C1C9H	SETDEV	—
NEW	C1E3H	SGN	C2D0H
NEXT	C1D2H	SIN	C2C8H
NOT	C1DBH	SORGN	B0C6H
OFF	C201H	SPACE$	—
ON	C1EBH	SQR	C2C0H
OPN	C1F9H	STATUS	C2E1H
OR	C1F2H	STEP	C2ECH
OUTSTAT	—	STOP	C2B7H
PAUSE	C24AH	STR$	C2D8H
PEEK	C224H	TAB	B0D0H
PEEK #	C21AH	TAN	C2FEH
PI	C213H	TERMINAL	—
POINT	C240H	TEST	B0D8H
POKE	C237H	TEXT	B0E1H
POKE #	C22DH	THEN	C2F5H
PRINT	C209H	TIME	C306H
RADIAN	C2A4H	TO	C322H
RANDOM	C28EH	TRANSMIT	—
READ	C271H	TROFF	C318H
REM	C2AFH	TRON	C30FH
RESTORE	C27CH	UNLOCK	C333H
RETURN	C266H	USING	C329H
RIGHT$	C299H	VAL	C33EH
RINKEY$	—	WAIT	C346H
RLINE	B0B1H	ZONE	—
RMT	B87DH		
RND	C286H		

6. táblázat. Üzenetek

NEW0?	C34FH
:CHECK	C355H
BREAK	C35CH
IN	C362H
ERROR	C36CH

PTA—4000

Forgástestek és felületek

A program a síkbeli koordinátarendszerben megadott paraméteres függvényt forgatja meg az Y tengely körül, és ábrázolja a megadott rálátással.

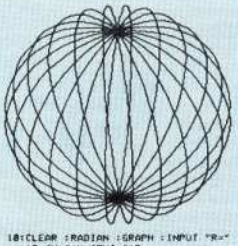
A program felépítése

10—16	kezdőértékek beállítása R: nagyítás N: szeletek száma PHI: a rálátás szöge COLOR 1, 2: színek (két szín is lehet) VÁLTÁS: két szín esetén
20—25	X( ) és Y( ) vektorok kiszámítása, X szélsőérték-kikeresés X( ) vektor transzformációja, Y( ) szélsőértékek kikeresése
30—60	forgatás és ábrázolás: külső ciklus: szeletekre bontás belső ciklus: egy szelet ábrázolása
500—520	A paraméteres függvény Q segédváltozóval, melynek értéke 0-tól 2π-ig fut.

Az *1.* és *2. ábrán* példaként néhány trigonometrikus függvényt ábrázoltunk.

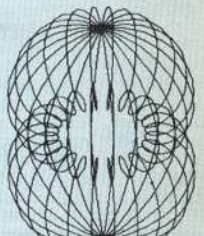
MÉSZÁROS CSABA

1. ábra



```
10:CLEAR :RADIAN :GRAPH :INPUT "R=";R,"N=";N,"PHI=";F
15:DIM X(60),Y(60):INPUT "COLOR 1=";K,"COLOR 2=";L,"VALTAS (1:I 0:N)  ";M
16:C=COS F:S=SIN F:Y=999:X=-Y:H=X:N=N+(N=2)*INT (N/2)
20:FOR I=0TO 60:Q=R/30*I:GOSUB 500:NEXT I:FOR I=0TO 60:Y(I)=Y(I)*C:IF X(I)>HLET H=X(I)
22:NEXT I:H=1/H:FOR I=0TO 60:X(I)=X(I)*H:U=Y(I)-X(I)*S:IF Y>ULET Y=U
23:IF K<ULET K=U
25:NEXT I:GLCURSOR (10,-K*15):SORGN
30:FOR I=.5TO N-1:Q=R/N*I:IF M=1GND I=N/2+.5LET J=K:K=L:L=J
40:U=SIN Q:D=COS Q*S:COLOR L
50:FOR J=0TO 60:Z=X(J)*U:U=Y(J)-X(J)*D:D=Z:IF J=0GLCURSOR (Z,U)
55:IF J=1GCOLOR K
56:IF J=46COLOR L
60:LINE -(Z,U):NEXT J:NEXT I:GLCURSOR (-10,Y-15):TEXT :END
100:REM
200:REM
300:REM
400:REM
500:X(I)=R*COS Q
510:Y(I)=R*SIN Q
520:RETURN
```

2. ábra



```
500:Y(I)=R*(COS Q+COS (2*Q)/2)
510:X(I)=R*(SIN Q+SIN (2*Q)/2)
520:RETURN
```

MEGRENDELHETŐ!

1.	MÓDSZERTANI SEGÉDANYAG A MUNKÁLTATÓK SZOCIÁLIS TERVEZÉSÉHEZ	300,— Ft
2.	LÍZING ALKALMAZÁSÁNAK GYAKORLATA	1200,— Ft
3.	BELKERESKEDELMI DOLGOZÓK BÉRBESOROLÁSA	900,— Ft
4.	VAGYONBIZTONSÁG A KERESKEDELEMBEN	500,— Ft
5.	KÖZÚTI ÁRUFUVAROZÁS I—II—III.	800,— Ft
6.	JÖVEDELEMÉRDEKELTSÉG A BOLTI KISKERESKEDELEMBEN	200,— Ft
7.	SZERZŐDÉSES ÜZLETEK NYILVÁNTARTÁSI RENDSZERE	300,— Ft
8.	KERESKEDELMI VÁLLALATOK KÖZÉPTÁVÚ TERVEZÉSE	930,— Ft
9.	SZERZŐDÉSES ÜZLETI VÁLLALKOZÁS A VENDÉGLÁTÁSBAN	300,— Ft
10.	FIZIKAI DOLGOZÓK BÉRBESOROLÁSA	600,— Ft
11.	ÜZLETI MUNKAREND MINTASZABÁLYZATA	300,— Ft
12.	A KERESKEDELMI VÁLTÓ ÉS HITEL GYAKORLATA	1500,— Ft
13.	A PÉNZTÁRI MUNKAFELTÉTELEK FEJLESZTÉSI IGÉNYEI ÉS LEHETŐSÉGEI	85,— Ft

KERESKEDELMI SZERVEZÉSI INTÉZET

Budapest XIII., Dózsa György út 150. 1134

Marketing osztály Telefon: 202-650, 202-670

A SYSTEM

Szervezési Vállalat
Budapest, Szent István krt. 11. 1055
Levélcím: 1363 Bp. 502., Pf. 33.
Telefon: 126-670

szisztémái — mikroszámítógépre

A felhasználói programok közül a SYSTEM Szervezési Vállalat HSZR—MICRO programcsomagját feltűnően sok, több mint 100, a KFR—MICRO és PEGA—MICRO programcsomagját 10-10 vállalat vásárolta meg.

A közelmúltban a HSZR—MICRO programcsomagért a vállalat az SZKI—SCITEL **partnertalálkozón I.,** a Software '86 **kiállításon és vásáron II. díjat kapott.** Feltehető a kérdés, hogy ez a köztudottan nagy számítógépet üzemeltető szervezési vállalat a **mikroszámítógépek alkalmazása és felhasználása** területén is miért tudott ilyen **jelentős eredményt** elérni, milyen **innovációs** képességekkel rendelkezik. A kérdésre a következőkben adunk választ.

A SYSTEM közel húsz éve foglalkozik beruházásszervezéssel. Állíthatjuk, hogy ez idő alatt a SYSTEM — beruházásszervezési **tapasztalatai** alapján — a beruházások szervezésének **szakértőjévé** fejlődött. Kezdetben a vállalat beruházásszervezési munkáját — érthetően — saját nagy számítógépével támogatta. Később a lehetőségek felismerésével, valamint az alkalmazók és felhasználók igényére, áttért a sok tekintetben **nagyobb rugalmasságot biztosító mikroszámítógépek** alkalmazására, és egymás után dolgozta ki a **HSZR—MICRO, hálószerkesztés és rajzoló sornyomtatóra, kombinált MPM/CPM módszer, vonalas ütemterv változtatható időtengellyel, erőforrásterv, aggregáció-histogram; a KFR—MICRO, a fejlesztések, a beruházások, a fővállalkozások sikeres irányítását elősegítő költségfigyelési; a PEGA—MICRO, a beruházások pénzügyi és gazdasági elemzését szolgáló stb. programcsomagokat.**

A SYSTEM a **H. B. Maynard and Ltd. angol szervező cégtől** megvásárolta az UNIVERSAL MAINTENANCE STANDARDS néven ismert UMS munkamérésen alapuló **karbantartási-irányítási rendszert.** A módszer segítséget nyújt a karbantartás valamennyi területének, a munkamegrendelési és elszámolási tevékenységnek, a megelőző karbantartásnak, az ösztönzésnek, az ellenőrzésnek a hatékony szervezésére. A SYSTEM a vállalati alkalmazás manuális munkaigényességének csökkentésére kifejlesztette **a mikroszámítógépes MICRO—UMS karbantartási munkalapkezelő és vezetői információt szolgáltató rendszert.** A rendszer bevezetése után a karbantartási munka termelékenysége minimum 40%-kal nőtt.

A munkatanulmányozás, a munkaszervezés, a munkatervezés közismert, de kevésbé használt módszere az MTM. A módszer alkalmazásával még a munka megkezdése előtt megállapítható a munka minden elemének kivitelezési módja, az ehhez tartozó idő és ennek megfelelően az optimális munkamódszer. Az előbbihez képest tömeges, nagyméretű, idő- és munkaigényes feladatok (termelő folyamatok) széles körű racionalizálását egy továbbfejlesztett változat, a H. B. Maynard cég által kidolgozott MOST rendszer teszi lehetővé. **A mikroszámítógépre kifejlesztett VIA—SYSTEM** varrodai **időadatrendszer segítségével — az elemzések alapján — gyors és megbízható normák képezhetők** és előkalkulációk készíthetők az igény szerinti konfekcionálási tevékenységre.

A SYSTEM számítógépes **anyaggazdálkodási rendszere a mikrogépes felhasználók szinte minden** nyilvántartási **igényét kielégíti,** mivel rendkívül rugalmasan képes alkalmazkodni a helyi sajátosságokhoz, igényekhez.

A SYSTEM állóeszköz-nyilvántartási és -elszámolási rendszere nemcsak a komplex állóeszköz-gazdálkodási rendszernek, de az üzemfenntartásnak is bázisát képezi. **Az állóeszköz-állományban és -értékesítésben bekövetkező változások mikrogépes kimutatása,** az értékcsökkenési leírások kiszámítása és elszámolása mellett ugyanis az állóeszközök műszaki adatainak nyilvántartását is biztosítja.

A SYSTEM **mikrogépes munkaügyi nyilvántartási rendszere** magában foglalja a személyzeti, oktatási, létszámgazdálkodási feladatköröket. **Az operatív irányítás feladatainak megoldásához szükséges** mikrogépes **információk** biztosítása elősegíti a gazdálkodás által megkívánt élőmunka-hatékonysági szint elérését.

A SYSTEM **mikroszámítógépes menetlevél-feldolgozási rendszere** biztosítja a gépjárművek **teljesítményadatainak** gyűjtését, az **üzemanyag-felhasználási elszámoltatások, fuvarköltségek** figyelését.

A SYSTEM **személyi számítógépre kidolgozott** vezetői rendszerével a **vezetői döntési alternatívák komplex hatékonyságát és rangsorát lehet megállapítani.**

Az SCR 11 MSZR számítógépekre készült szubrutincsomag a FORTRAN, a BASIC és C nyelvi **interaktív programok készítését segíti, több terminálkezelési lehetőséggel és minimális memóriaigénnyel.** A kidolgozott **QUICK—SYS programfejlesztő rendszerrel a mikroszámítógépes programok futási sebessége jelentősen növelhető.**

A mikroszámítógépek vállalati alkalmazása — többek között — az elektronika **befogadására alkalmas vállalati szervezetet** is igényel. A SYSTEM foglalkozik az erre alkalmas korszerű vállalati szervezet és vállalatirányítási rendszerek kidolgozásával, bevezetésével, az iparvállalatok működési és szervezeti rendszerének komplex átalakításával, a vállalati érdekeltségi rendszerek kidolgozásával, kialakításával, **a vezetési, irányítási információk mikroszámítógépes feldolgozásának előkészítésével.** A számítástechnikában rejlő lehetőségek kényelmes és szakszerű kihasználását szolgáltatja a vállalatok számára a SYSTEM számítóközpontja. IBM kompatíbilis nagy számítógépe — amelyhez bérletszerű formában kihelyezett terminálokat is a megbízók rendelkezésére bocsát — minden igényt kielégít. A SYSTEM számítóközpontja megbízói részére komplex kiszolgálást biztosít, amely magába foglalja a **konkrét gépi futtatások** elvégzésén kívül a megfelelő **számítógépes munka előkészítését,** valamint — az üzemeltetési dokumentációban rögzített szintig — az **eredménytáblák ellenőrzését,** esetenként a **kész eredménytáblák kiszállítását** is.

A SYSTEM számítóközpontja vállalja **TAF és osztott adatfeldolgozási rendszerek tervezését, fejlesztését, üzemeltetését.**

A SYSTEM a FERROGLOBUS, a METALLOGLOBUS, a VEGYTEK és a VILLÉRT TEK vállalatokkal létrehozta az **UNIglOb Készletinformációs Értékesítő Közös Vállalatot,** melynek célja a vállalati felesleges és lassan mozgó készletek mobilizálása, az anyaghiány miatti termeléskiesések kiküszöbölése, korszerű **számítástechnikai** eszközök segítségével.

Rövid ismertetőnkben természetesen lehetetlen a SYSTEM valamennyi módszerét, eljárását, szolgáltatását bemutatni. Abban viszont **biztosak vagyunk,** ha felkeresi a vállalatot, problémáira **megtalálja** a legjobb megoldást.

TURBO interfész ZX-Spectrumhoz

Nemrég mutattak nekem egy gyári készítésű TURBO interfészt. Amikor elmondták, mit tud, elhatároztam, hogy én is készítek egy hasonlót. Nosza, szétszedtem, de a sejtésem beigazolódott: az áramkörben egy „Taiwan" feliratú, speciális integrált áramkör volt. Ettől egy kissé elment a kedvem. Később mégis gondolkoztam, hogy mit lehetne tenni, és akkor jutott az eszembe a következő megoldás.

Szükség van egy IC-re, aminek 10-11 bemenete és 8 kimenete van. A kimenetek állapota mindig a bemenetek állapota szerint változik. Ezt a funkciót egy kellően beprogramozott EPROM is ellátja. A jól bevált i2716 típus mellett döntöttem, mivel ezt könnyen be lehet szerezni, és az ára is elfogadható.

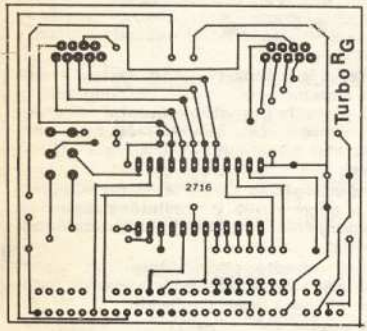
A TURBO ismeri a háromféle botkormány-illesztési módot, a Kempston, a Protek és a Sinclair Interface II üzemmódot.

— A Kempston az IN 31 utasítással olvassa be a botkormány bitjeleit, és így azonosítja az irányítást.

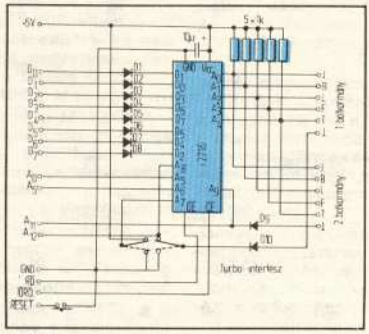
— A Protek a felső billentyűzetsort olvassa le, az irányítást a nyíllal jelzett billentyűk, a lövést a 0 jelenti.

— Az Interface II szintén a felső sort olvassa le, de ez két botkormányt kezel úgy, hogy egy fél sor jelent egy botkormányt.

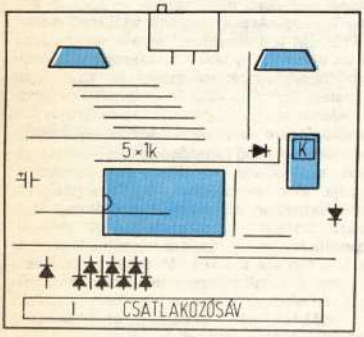
Ebből is látszik, hogy a Protek és az Interface II fedik egymást. Ezt a problémát egy kapcsoló beiktatásával oldottam meg. Ez a kapcsoló egyrészt Ø vagy 1 állapotba hozza az EPROM egyik bemenetét, másrészt az 1. számú (jobb oldali) botkormány közös pontját kapcsolja a földre vagy az A12 vezetékre.



1. ábra



3. ábra



2. ábra

A kapcsoló felső állásában az 1. számú botkormány Kempston és Protek üzemmódban működik, a kapcsoló másik állásában a két botkormányt Interface II módon illeszti az áramkör.

Ezenkívül a bemenetekre csatlakozik még az AØ, ami a billentyűzetet címezi, az A5, ami a Kempstont címezi, az A11 és az A12, ami a billentyűzeten a két felső fél sort választja ki. A CE bemenetre kapcsolt IORQ és az OE-re kapcsolt RD gondoskodik arról, hogy az EPROM csak az IN műveletekre aktivizálódjon.

A 8 db dióda (D1—D8) biztosítja, hogy az áramkör ne tiltsa le a billentyűzetet. A diódák típusa 1N4148 vagy 1N914.

Az EPROM-ot érdemes foglalatba tenni, mert egyébként a beforrasztáskor tönkremehet. A botkormány-csatlakozókat kb. 10 mm hosszú ónozott rézhuzalok beforrasztásával lehet helyettesíteni, melyeknek az átmérője 1 mm. A kapcsolás RESET gombjául a NYÁK tetejére szerelt mikrokapcsoló szolgál.

A kapcsolás helyes működéséhez az EPROM-ba a lista szerint közölt programot kell beégetni. Mivel csak 1 kbájt memóriát használunk fel, az EPROM A10 bemenetét földre kell kötni, így csak az első 1 kbájtot kell beégetni. A listában nem közölt bájtokba FF (255) tartalmat kell beégetni.

Az utánépítők munkájának megkönnyítésére közlöm a fóliarajzot *(1. ábra)*, a beültetési rajzot *(2. ábra)* és a kapcsolási rajzot *(3. ábra)*. A NYÁK elkészíthető egyoldalas kivitelben átkötésekkel, illetve kétoldalas kivitelben. A közölt dokumentációk alapján, megfelelő felszereltséggel az interfész könnyen elkészíthető. Ha valakinek mégsincs lehetősége a NYÁK elkészítésére vagy az EPROM beégetésére, annak készséggel segítségére lesz e sorok írója.

RUSZNYÁK GÁBOR

EPROM program

```
0020  CB C9 CA C8 C3 C1 C2 C0
0028  8B 89 8A 88 83 81 82 80
0030  4B 49 4A 48 43 41 42 40
0038  0B 09 0A 08 03 01 02 00

0120  CB C9 CA C8 C3 C1 C2 C0
0128  8B 89 8A 88 83 81 82 80
0130  4B 49 4A 48 43 41 42 40
0138  0B 09 0A 08 03 01 02 00

0140  7F 7F FF FF 7F 7F FF FF
0148  7F 7F FF FF 7F 7F FF FF
0150  7F 7F FF FF 7F 7F FF FF
0158  7F 7F FF FF 7F 7F FF FF

01C0  34 35 36 37 3C 3D 3E 3F
01C8  74 75 76 77 7C 7D 7E 7F
01D0  B4 B5 B6 B7 BC BD BE BF
01D8  F4 F5 F6 F7 FC FD FE FF

0220  CB C9 CA C8 C3 C1 C2 C0
0228  8B 89 8A 88 83 81 82 80
0230  4B 49 4A 48 43 41 42 40
0238  0B 09 0A 08 03 01 02 00

0240  35 3D 35 3D B5 BD B5 BD
0248  75 7D 75 7D F5 FD F5 FD
0250  37 3F 37 3F B7 BF B7 BF
0258  77 7F 77 7F F7 FF F7 FF

02C0  34 74 B4 F4 3C 7C BC FC
02C8  35 75 B5 F5 3D 7D BD FD
02D0  36 76 B6 F6 3E 7E BE FE
02D8  37 77 B7 F7 3F 7F BF FF

0320  CB C9 CA C8 C3 C1 C2 C0
0328  8B 89 8A 88 83 81 82 80
0330  4B 49 4A 48 43 41 42 40
0338  0B 09 0A 08 03 01 02 00
```

Ki ad magyarázatot?

Szieberth András olvasónk SPECTRUM gépén tapasztalta a következőket:

Ha a DRAW utasítás után harmadik számként

300—1000

közti számot adunk meg, akkor egyes esetekben hibaüzenetet kapunk, más esetekben érdekes alakzatok jelennek meg a képernyőn.

Legyen pl.

PLOT 130,80:DRAW 100,30,1400

Az eredmény egy háromszög, amely forog. Az utolsó szám helyett 1900

értéket adva ötszöget kapunk.

A jelenség magyarázatát szakkönyvekben sem találta meg.

BITEK ÉS FIGURÁK

ÁLLÁSÉRTÉKELÉS IV.

Királytámadás és védelem I.

Az értékelőfüggvény egyik nagyon fontos komponense a királytámadás és a királyvédelem értékének a meghatározása, hiszen tulajdonképpen az egész sakkjáték ezen alapszik. Az előbbiekben általános sakkstratégiai elemeket ismertettünk, most pedig a kitüntetett szereppel bíró figura, a király támadásának és védelmének kvantitatív meghatározását tárgyaljuk. Az előbbi cikkben taglaltuk, hogy a középen rekedt király mennyire hátrányos, mert kettéosztja saját haderejét, megakadályozza annak ésszerű elhelyezését és átcsoportosítását. A középen rekedt királyra az ellenfél könnyen csapást mérhet, mert miután a megnyitásban a centrumgyalogokkal kiléptünk helyükről, a középen maradt királyt a gyalogok nem védik, és a tábla közepén megnyíló vonalakon, átlókon keresztül hamar az ellenséges figurák célpontjává válik. Ezért szinte nélkülözhetetlen a sakkjátékban a mielőbbi sáncolás.

Steinitz még gúnyolódott azokon, akik királyukat szerinte gyáván sáncolással a gyalogok mögé rejtették. Az ő királya — mondotta — bátran rohan a harcba. De már a kortársai között is akadtak, akik a király biztonságáról való gondoskodást — a sáncolást — nem a gyávaság, hanem a körültekintés és az óvatosság megnyilvánulásának tartották. A romantikus sakkiskola hívei is kevesebb figyelmet szenteltek a sáncolásnak. Az akkor divatos megnyitások — királycsel, Evans-csel — gyakran jártak a sáncolás korai elvesztésével.

A mai gyakorlatiasságra hajló korban azonban, amikor a versenyszerű sakkozás térhódításával az eredményesség szempontja mindenek felettivé vált, sokan nem gondolnak aktív hadműveletre addig, amíg saját királyuk biztonságát sáncolás útján meg nem teremtették. Az egészséges játszmafelépítés elengedhetetlen feltételének tekintik, hogy a király és a bástya közti könnyű tiszteket mielőbb kifejlesszék, hogy azok ne akadályozzák a sáncolást.

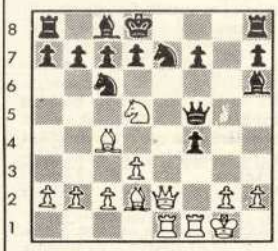
Akadnak kivételek, amikor a sáncolás előtt hasznosabb lépéseket is tehetünk. Vannak megnyitási változatok, amelyekben csak a tizenötödik, amelyekben a huszadik lépés között kerül sor a sáncolásra. A sáncolás lehetőségét azonban, még ha nem is élünk vele, azonnal célszerű mindig készenlétben tartani.

A játszmák többségében rövid sáncot alkalmaznak. Ehhez csak két bábuval, a huszárral és a futóval kell lépni, míg a hosszú sánc esetén a vezérnek is el kell hagynia alapállásbeli mezejét. A futó és a huszár korán fejleszthető figurák, a vezért viszont később kell fejleszteni, mert ellenkező esetben a könnyű tisztek célpontjává válhat, és tempóvesztésekkel kell visszavonulnia.

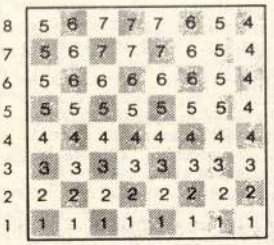
A rövid sáncot más szempontból is biztonságosabbnak tartják a hosszú sáncolásnál. Rövid sánc esetében a királyt az f2, g2, h2 gyalogok védik, míg hosszú sánc esetében, amikor a király c1-re kerül, annak a d2, c2, b2 gyalogok felelnek meg. De a vezérfutó lépése előtt a d és b gyalogok közül legalább az egyikkel el kellett lépnünk, hogy így lehetővé váljék a hosszú sáncolás. A királyt védő gyalogosok közül tehát egyiknek el kellett lépnie a helyéről, s így a királyállás eleve meggyengült. Ezenkívül a védtelen a-vonal és az a2 (a7) gyalog hosszú sánc esetén további támadható pontot jelent az ellenfélnek. Ezért a királyállás biztonságosabbá tétele érdekében szükség van még a Kc1—b1 lépés megtételére is. Ennek azonban hátránya, hogy a király már a megnyitásban kétszer lép, és ez tempóvesztést jelent.

Rövid sánc elleni támadásnál két eset között kell különbséget tennünk.

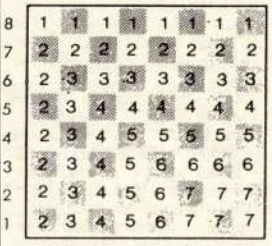
1. Támadás meggyengített királyállás ellen, amelynél a királyállást védő három gyalog egyike már lépett vagy eltűnt a tábláról.



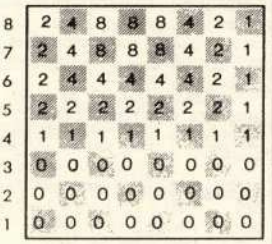
1. ábra



2. ábra



3. ábra



4. ábra

2. Támadás intakt, érintetlen királyállás ellen, ahol a gyalogok még eredeti helyükön állnak.

A két eset egyébként elég gyakran átmegy egymásba, vagyis az intakt királyállást a védekező az ellenfél különböző fenyegetéseinek hatása alatt kénytelen valamely gyaloglépéssel meggyengíteni. Ebből a szempontból legfontosabb az f3 (f6) huszár szerepe, amelyet a rövid sánc legfontosabb védőjének tekinthetünk. Ha ezt sikerül valamely módon, csere, gyaloggal való támadás, esetleg áldozat útján eltávolítanunk, a h7 pontra gyakorolt nyomással vagy más fenyegetésekkel rendszerint sikerül a g vagy h gyalog lépését kikényszeríteni. Mennyiben bomlik meg az állás egyensúlya e lépések által, amelyek látszólag csak minimálisan módosítják a gyalogállás szerkezetét?

Valójában többféle okból is lényegesen gyengíthetik a hadállást. Ezek a gyaloglépések megkönnyíthetik a vonalnyitást a királlyal szemben, akár oly módon, hogy az ellenfél előrenyomuló gyalogjai cserélődnek e gyalogok ellenében, ami a g- vagy h-vonal megnyitására vezet, akár esetleg tiszteláldozat árán is. Másrészt a gyalogok által védtelenül hagyott mezőkön át az ellenfél tisztjei a király közvetlen közelébe kerülhetnek. Különösen világos ez például g7—g6 (g2—g3) lépés esetén. A védtelenül maradt h6 (h3) mezőn, sőt többnyire az f6 mezőn is ellenséges tisztek jelenhetnek meg, és fokozhatják a támadást.

Ezeket a szempontokat a sakkprogramnak is figyelembe kell vennie. De mielőtt ezzel konkrétabban foglalkoznánk, tanulmányozzuk a két fél ellentétes oldalra történő sáncolásának esetét!

A támadás legszebb példáival találkozhatunk az ilyen játszmákban. Ez érthető, hiszen az egyensúly — a két király elhelyezkedésének szempontjából — erősen megbomlott. Az azonos oldalra történő sáncolásnál a királyok szimmetrikus elhelyezkedése sokáig egyensúlyban tarthatja az állást. Az ellentétes oldalra történő sáncolásnál viszont nyugodtan támadhatunk minden eszközzel az ellenfél királyának térségében, mert ezek az akciók nem járnak egyidejűleg a saját királyállásunk veszélyeztetésével. Ezért itt a támadás egyik leggyakoribb módszere a gyalogroham. Ennek célja itt is a vonalnyitás az el-

lenfél királyával szemben. Sok esetben a gyalogrohamot nem is lehet passzív védekezéssel elhárítani. Ezért szokták mondani, hogy ellentétes oldalú sáncolás esetén a legjobb védekezés az ellentámadás. A küzdelem a végsőkig kiéleződik, igen sok szerephez jut az idő, minden tempónak döntő jelentősége lehet. Ezért a merész áldozatok, meglepő taktikai fordulatok egész tárházát alkalmazhatja a támadó, mert ha a számára fontos hadszíntéren kierőszakolja a döntést, nincs jelentősége annak, hogy a másik hadszíntéren esetleg anyagi vagy pozíciós hátrányban marad. Itt valóban érvényes az a tétel, hogy a döntő időben a döntő hadszíntéren kell előnyre szert tenni.

Mivel ilyen jellegű küzdelem során az anyagi egyensúly is gyakran felborul, konkrét, változatonkénti számítás szükséges, aminek a számítógép gyorsan és pontosan eleget tud tenni.

Aljechin a következőket mondta: „A két szárnyon egyszerre folyó hadművelet az én kedvelt stratégiám. Az ilyen játszmáim fő jellegzetessége a váratlan és gyors döntést hozó támadás. A támadásokat nem a célpont közelében készítem elő, hanem valamennyi előkészítő manőver, melynek célja a királyállást védő figurák eltávolítása volt, a centrumban vagy az ellenkező szárnyon bontakozott ki. Igen érdekes az a körülmény is, hogy a döntő kombináció, melyet pörölycsapáshoz lehet hasonlítani, a futó támogatásával és mindig áldozattal történt."

Ozsváth András mesteredző így foglalt állást: „Az aszimmetrikus királyállások a játék különböző fázisaiban éreztetik hatásukat, és ezzel hozzájárulnak a hadállások kiélezéséhez. Ezért a különböző oldalú sáncolás, mint az állás egyensúlyának megbontására irányuló törekvés, szinte kísérő jelensége a modern megnyitáskezelésnek is. Aki győzelemre tör, az kockáztat, feloldítja a szimmetriát, megbontja az egyensúlyt, és ellenkező oldalra sáncol."

Ezeket az elveket figyelembe véve célszerű sakkprogramunkat is elkészíteni. Tudvalevő, hogy a programoknak a végjáték a gyengéje, ezért úgy kell vezetnünk a játékot, hogy lehetőleg már a középjátékban eldőljön a játék kimenetele. Ezért David Levynek az előző számban említett példájával szemben az ellentétes oldalra történő sáncolás felpontozását tartom jónak, annak ellenére, hogy ezekben az esetekben legtöbbször a programnak kell a vezérszárnyra sáncolnia. A programnak fel kell mérnie a király megtámadásának mértékét, aminek kvantitatív meghatározása nagyon nehéz feladat. Bemutatok egy elég jónak bizonyuló közelítést.

Az ellenséges király elleni támadás erősségének meghatározására egyszerű módszer a király körüli mezők súlyozása. Hasonló, mint ahogy az előző cikkeinkben láttuk a centrum ellenőrzésekor.

A király melletti mezőkhöz sokkal nagyobb pontértéket rendelünk, mint a távolabbiakhoz. Így a király körül a királytól távolodva koncentrikus körökben egyre kisebb lesz a táblamező súlyozásának mértéke. A program támadás esetén összesze tudja hasonlítani az értékeket, és ebből kiderül hogy mekkora aktivitást fejt ki.

David Levy egy egyszerű terv alapján 7-tel számolta a király által elfoglalt és azzal határos mezőket, 6-tal az ezeket határoló mezőket, 5-tel az ezután következőket és így tovább. Az *1. ábrán* bemutatott hadállás királytámadási táblája a fentiek alapján világos szempontjából a *2. ábrán*, sötét szempontjából a *3. ábrán* látható módon alakul.

Ha összeszámoljuk az egyes figurákra vonatkozó pontértékeket, a következő eredményre jutunk.

A világos figurákra:

Be1:	1+1+1+1+1+2=7
Bf1:	1+1+2+3+4=11
Kg1:	1+1+2+2+2=8
a2:	3
b2:	3+3=6
c2:	3+3=6
Fd2:	1+1+3+4+5+3+4 =21
Ve2:	1+1+1+2+2+3+4 +5+6+7+3+4+4= 43
g2:	3+3=6
h2:	3
d3:	4+4=8
Fc4:	2+3+3+5+6+5=24
Hd5:	3+3+4+4+6+6+7- +7=40

A sötét figurákra:

Ba8:	1+2=3
Fc8:	2+2=4
Kd8:	1+1+2+2+2=8
Bh8:	1+1+1+1+2=6
a7:	3
b7:	2+3=5
c7:	3+3=6
d7:	3+3=6
He7:	1+1+3+3+4+4=16
f7:	3+3=6
h7:	3
Hc6:	1+1+2+2+2+4+3- +5=20
Fh6:	1+2+4+5=12
Vf5:	2+3+2+3+2+3+4- +4+4+4+5+5+5+- 5+6=57
f4:	6+6=12

Így világos figurái számára a királytámadási érték 186, a sötétekre pedig 167.

Láthatjuk, hogy a különbség világos figuráinak javára mutatkozik, de nem egészen tükrözi a támadás erősségét, mely a sötét királyra sokkal veszélyesebb, mint amit a 186 és 167 pontérték közötti különbség mutat. Ennek oka, hogy itt bizonyos mezők pontértékeit még nem vettük figyelembe.

Észre kell venni, hogy vannak figurák, melyek egy másikon keresztül is kifejtik hatásukat. Ilyen például az e1 bástya. Ez az e2 mezőn álló vezéren túl hatva fejti ki hatását a sötét térfelére. Ha ezeket a mezőket is hozzáadjuk a támadás mértékéhez — ami még +27 pontot jelent világosnak —, akkor az eredmény még feltűnőbben világos számára kedvez. Ezenkívül az e1 bástya a g1 mezőre is hat f1-en keresztül, és ugyanígy korrigálható az f1 bástya királytámadási értéke is. Ezek után a királytámadás valódi értéke: 221 világos és 167 sötét számára. Ez hívebben tükrözi a két fél támadási értékét.

Még jobb közelítés adható ennek az értéknek a megállapítására, ha a királytól túl távoli mezőknek 0 pontértéket adunk — mivel ezek a mezők a királytámadás szempontjából kevéssé játszanak szerepet —, és az így súlyozott táblát veszszük figyelembe. Ugyanerre az állásra a sötét király szempontjából ez a *4. ábrán* látható módon alakul.

A királytámadás szempontjából még sok mást is figyelembe kell vennünk, de ennek ismertetésével a következő folytatásban foglalkozunk.

KOVÁCS P. ATTILA

K. L. Boon rajza

```
680 PM=PM+1:IFPM=5THEN PM=1
```

28. lista

```
690 IFX>MX-20ANDX<MX+6ANDY<MY+13
ANDY>MY-10THEN MF=0:LINE(MX,MY)-
(MX+5,MY+11),PRESET,BF:SC=SC+10:
PLAY"T25504V31GFGDED":FU=FU+10:M
C=MC+1
```

29. lista

```
700 IFMC>=5THEN LV=LV+1:MS=MS+1:
GOTO2240
```

30. lista

```
2240 BONUS=(LV-1)*100:SC=SC+BONU
S
2250 CLS:PRINT@128,"WELL DONE, Y
OU HAVE FINISHED      LEVEL";LV-1
2260 PRINT@234,"BONUS=";BONUS
2270 TUNE$="02L4GG;L2GDL4BBL2BGL
4GBG3L2DDL4C02BL1AL4AB03L2CC02L4
BAL2BGL4GBL2ADL4F#AL1G;"
2280 ZZ$=TUNE$+TUNE$:PLAY"T6"+ZZ
$
2290 GOTO2160
```

31. lista

```
710 PSET(RND(256),RND(70)+22)
720 GOSUB920
```

32. lista

```
920 IFFU>190THEN FU=190
930 FU=FU-.75
940 LINE(FU,3)-(200,10),PRESET,B
F:LINE<FU-1,4)-(30,9),PSET,BF
950 IFFU<31THEN1270
960 IFFU<90THEN SOUND150,1:PUT(1
40,3)-(178,11),DA,PSET
```

33. lista

```
1270 PUT(X-3,Y-2)-(X+20,Y+14),CR
,PSET
1280 SOUND200-Y,1
1290 Y=Y+2
1300 PUT(X,Y-2)-(X+23,Y+14),CM,P
SET
1310 SOUND200-Y,1
1320 PUT(X,Y-2)-(X+23,Y+14),CL,P
SET
1330 SOUND200-Y,1
1340 Y=Y+2
1345 IFY<150THEN1270
1350 PUT(X,Y)-(X+20,Y+11),CR,PSE
T:FORT=3TO10STEP-2:PLAY"V"+STR$
(T)+"T255;O3;DEAD":NEXT:FORT=1T
0500:NEXTT
```

34. lista

35/a lista

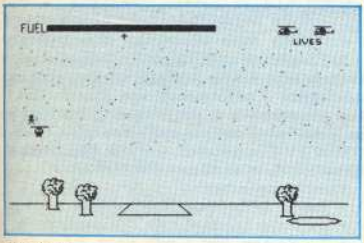
```
970 GOSUB490
980 RETURN
490 IFL1=1THEN550
500 X1=200
510 ON PC GOSUB2100,2120,2140
520 IFL1=2THEN550
530 X1=227
540 ON PC GOSUB2100,2120,2140
550 PC=PC+1:IFPC=4THEN PC=1
560 RETURN
```

```
2100 PUT(X1,1)-(X1+23,14),CR,PSE
T
2110 RETURN
2120 PUT(X1,1)-(X1+23,14),CL,PSE
T
2130 RETURN
2140 PUT(X1,1)-(X1+23,14),CM,PSE
T
2150 RETURN
```

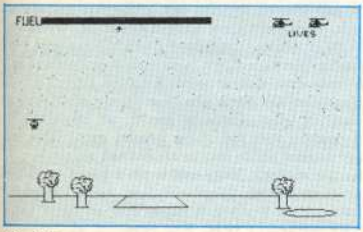
35/b lista

A játékprogramozás technikája

A FŐCIKLUS



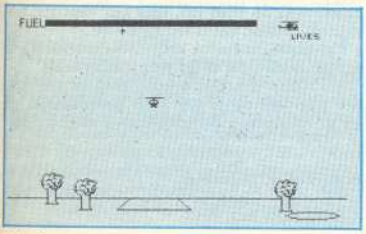
27. ábra



28. ábra

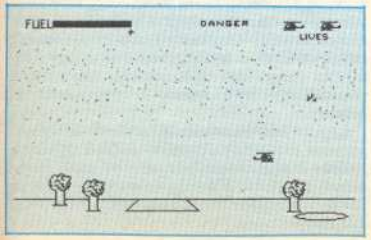
```
WELL DONE, YOU HAVE FINISHED
 LEVEL-1


     BONUS= 0
```

29. ábra



30. ábra

31. ábra



Az emberkék zuhanása

A *28. listában* a játék egyik fő része látható, melyben ciklikusan változtatjuk a zuhanó emberke pozícióját. Az emberke zuhanás közben hol áll, hol fekszik, hol pedig fejjel lefelé helyzetben van, és eközben zuhan.

Emlékeztetőül: a fejjel lefelé helyzetnek az M változó (PM = 1), az álló helyzetű emberkének az M2 változó (PM = 2;4), a fekvő helyzetű emberkének az M1 változó (PM = 3) felel meg.

A pontszerzés lehetősége

Ha sikerült időben elkapnunk a zuhanó emberkét, pontot szerezhetünk, amennyiben mind az ötöt elkapjuk, úgy még jutalompont is jár. Fontos része tehát a programnak annak vizsgálata, hogy sikerült-e elkapnunk? Ez a *29. listában* leírt programrészlettel történik. Ha igen, az emberke „eltűnik", vagyis a helyét egy üres négyszöggel töltjük fel; kapunk 10 pontot, zenét, többletenergiát (FU); nő az elkapott emberkék száma (MC).

Megjegyzendő, hogy az „elkapáshoz" nem szükséges a helikopter és az emberke tényleges találkozása, elég, ha erősen megközelítjük. Ez látható két pozícióban, a *27.* és a *28. ábrán.* Az ezek előállításához szükséges programmódosítás:

690 ... THEN STOP: ... MC = MC + 1:STOP

A futtatás először

RUN

majd

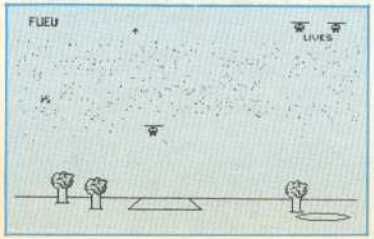
CONT

A dallam egy rövid, gyors futam, amelynek hangja olyan, mint amikor valaki csücsörített szájjal beszívja a levegőt.

Mindet elkaptuk?

A *30. listában* megvizsgáljuk, hogy mind az 5 emberkét sikerült-e elkapnunk. Ha igen, nő a szintszám (LV), a bonyolultsági mutató (MS), és rátérünk egy mellékágra.

32. ábra



A „siker" mellékága

Ha mind az 5 emberkét elkaptuk, akkor egy mellékágban különböző jutalmazásokat teszünk. Ez úgy történik, hogy először kiszámítja a programrészlet az elért pontszámot, törli a képernyőt, kiír egy dicséretet, jelzi, hogy melyik szinten sikerült túljutnunk, majd eljátszik egy jutalomzenét. A kijelzett dicséretet a *29. ábra* mutatja. Mindezt a *31. listában* látható programrészlet valósítja meg.

A programmódosítások:

690 visszamódosítani

2285 STOP

A futtatás:

RUN 2240

A mellékág folytatása az eredményjelzésen keresztül történik. Az eljátszott jutalomzene az „Óh, te drága Clementina" című dalocska.

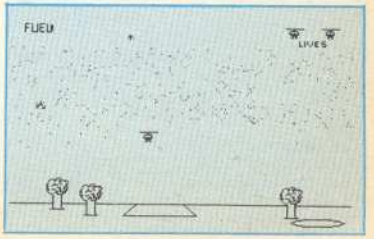
A közös főág

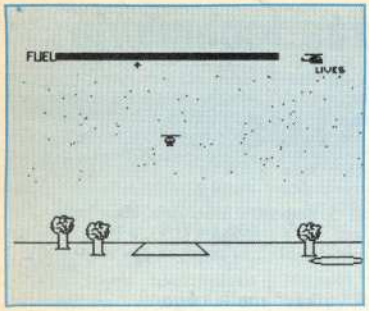
A *32. listában* a 640-es sortól ugrás és az előző főág közös folytatása látható.

Üzemanyag-vizsgálat

A játék során az üzemanyag fogy, töltéskor pedig egy bizonyos szintig növekszik. Az üzemanyag mennyiségével kapcsolatosakat a *33. listában* lévő programrészlet intézi. A 920-as sor megakadályozza a „túltöltődést". A következő sor fogyasztja az üzemanyagot, és egyúttal ábrázolja annak mennyiségét. Ez a *30. ábrán* látható. A 950-es sor vizsgálja, hogy elfogyott-e az üzemanyagunk. Ha igen, rátér egy mellékágra, ha még nem, de már veszélyesen kevés, akkor vészjelzést ad (960-as sor és *31. ábra).*

33. ábra





34. ábra

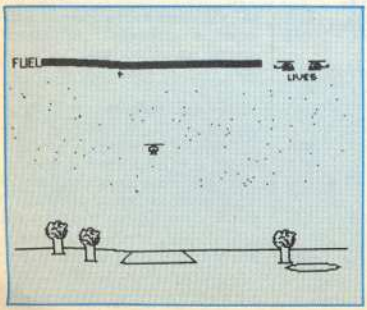
A szükséges módosítások:
2285 nincs
945 IF FU<110 THEN STOP
965 STOP
A futtatás:
RUN
majd
CONT
A hangeffektus egy éles, rövid zaj.

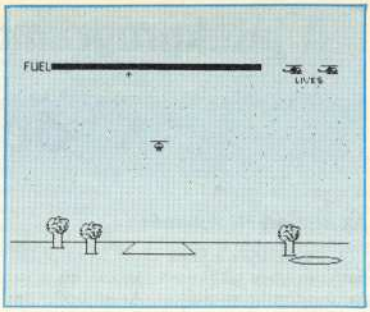
Az üzemanyag-kifogyás mellékága

Ha elfogyott az üzemanyagunk, helikopterünk lezuhan. Ezt a *34. lista* valósítja meg. Ezen belül váltakozva felrajzolódik a különböző irányba néző helikopter, egyre lejjebb zuhanva, egyre mélyülő hangú zörejjel. Amíg nem ér földet, ez ciklikusan ismétlődik. A földet érést gyorsan rezgő rövid dallam jelzi. Ha már lezuhant a helikopter, az 1480-as sorban ismertetettel azonos hanghatás, és az 1370-es sorral történő folytatás következik. A zuhanó helikoptert a *32.* és a *33. ábrán* láthatjuk.

A szükséges programmódosítások:
945, 965 nincs
1275 STOP
1305 STOP
1365 STOP
A futtatás:
RUN
majd
CONT

36. ábra





35. ábra

és ismét
CONT

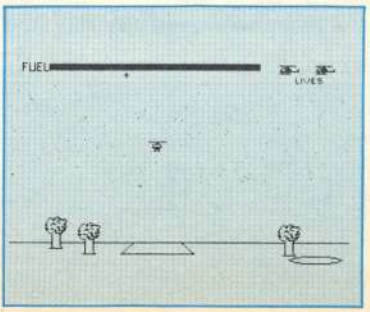
A főciklus folytatódik

A *35. listában* látható a főciklus ezen szubrutinjának még hátralevő része. Ebben az életszámtól függő helikopterhelyzet-rajzolás és néhány paraméter beállítása történik. Az ehhez tartozó rajzot a *34.* és *35. ábra* mutatja.
A futtatáshoz szükséges módosítások:
1275, 1305, 1365 nincs
515 STOP
545 STOP
A futtatás:
RUN
majd
CONT
PC=1
eset volt beadva
PC=2
értéket és
RUN 490
majd
CONT
után adódik a *36.* és *37. ábra*, melyeken látható, hogy a helikopterek a játék közben ide-oda forognak.

Záró cikkünkben a program még hátralevő részeit ismertetjük, és néhány tanácsot adunk az érdeklődőknek.

GALINA FERENC

37. ábra



Pallagi László, Százhalombatta,

Rózsa F. u. 3. 2440

Van egy MSX számítógépem, ami egy nagyszerű gép, használhatatlan gépkönyvvel. Szívesen leveleznék vagy személyesen találkoznék MSX típusú számítógépek tulajdonosaival. Főleg programokat és dokumentációt cserélnék.

Az MSX nem nagyon gyakori gép az országban, azért remélem, hogy találni fog egy-két hasonló problémával küszködő társat.

Láposi Gábor, Nyíregyháza,

Széchenyi út 24.

Rendszeres olvasója vagyok az önök által szerkesztett lapnak. Az eddig megjelent számok mindegyike a birtokomban van. (Szerencsére.) Már két számban is megjelent a „Micsoda kiszolgálás" című cikk. Az utóbbiban épp a VIDEOTON TV-Computeréről volt szó. Egyetértek a cikk írójával, ez az állapot SIRALMAS.

Ezzel kapcsolatban:

A minap beugrottam Nyíregyházán a Centrum áruházba. Legalább 3 db TV-Computert láttam a polcokon, illetve még egyet. És ez még nem minden. Ez az egy nem volt a pult mögött elbarikádozva, hanem egy külön kis asztalkán volt. Nem dobozban, hanem egy színes VIDEOTON tv tetején! Ott pihent (működött) a gép, barátságosan kacsingató cursorral. Kérdéseimre jóformán csak az árát tudták közölni. Ez egy kicsit elszomoritott. Bár a gép csak aznap jelent meg a vásárlók előtt. Érzésem szerint az elkövetkező napokban sem fog sokat javulni ez az állapot. Így viszont nehéz lesz az eladóknak meggyőzni a tisztelt vásárlót, hogy csakis ezt a gépet érdemes most megvenni. Megkérdeztem, hogy kipróbálhatom-e ezt a gépet. A válasz csupán ennyi volt: TERMÉSZETESEN. Egy egyszerű kis programot írtam, ami távozásom után továbbra is futott.

Ide tartozik még, hogy aznap igen meleg volt. Egy szál ingben, illetve nadrágban mentem be a boltba. Egyértelműen látszott, nincs nálam 20 ezer forint. Mindezek ellenére még a mellékelt „dokumentációt" is megkaptam. Nem tagadtam, hogy jelenleg rendelkezem egy Spectrum géppel. A közeljövőben nem akarok vásárolni sem ilyen, sem más gépet. Érdeklődve nézték, hogy mit csinálok. Hisz ha vevő jön, meg kell vagy inkább kellene mutatni, mit is tud ez a gép!

Egy biztos, az eladók kedvessége már súrolja a nyugati szintet. De ez így még mindig **kevés**.

A VIDEOTON TV-Computer-ének az ára közben megváltozott, 12 000 Ft-ra csökkentették, és így az elfogadható árszintet nagyon megközelítette.

Bogyó Gábor, Hajmáskér,

Kossuth Lajos út 1. 8192

Véleményem szerint Magyarországon nagyon hiányzik egy összefoglaló prospektus, amely tartalmazná a számítástechnikai, műszaki és egyéb cikkek aktuális vámtarifáját, amely a lakosság széles köreinek is hozzáférhető lenne. Egy ilyen sokrétű (megújuló) tájékoztatóval szerintem a vámszabálysértések száma is nagymértékben csökkenne.

Legjobb tudomásom szerint a Vámhivatal folyamatosan aktualizált díjalappal dolgozik. A hivatal listáján a Magyarországon népszerű gépek szerepelnek. Ha valaki a listába fel nem vett gépet kíván behozni, akkor ennek vámértékét szakértőkkel állapittatják meg.

Az olvasó írja

Olvasóink ismét nagyon sok jó gondolatot adtak. Köszönjük. Valamennyi olvasói levelet nyilvánvalóan nem tudjuk közölni, így a legérdekesebbekből válogattunk. Nagyon sokan kérnek — még külföldről is — gépkönyveket, forrásnyelvi programokat, ROM leírásokat, különféle perifériák főleg magyar nyelvű dokumentációit. Ezeket — sajnos — nem tudjuk megszerezni, ezért elküldeni sem. Ha közérdekű kérésről van szó, akkor a levelet közöljük, hátha valaki olvasóink közül segíteni tud.

Szabó László, Békés

Teleky u. 105. 5630

Amiért önhöz fordulok, az a következő: már sok szoftverleírást olvastam a lapban, s én szeretném, ha ez a szokás megmaradna. Két programról küldenék leírást. Nem akarom, hogy mások is úgy járjanak, mint én. Két hetembe került a programok megismerése. A programok a következők: Koalopainter és a Doodle. Mindkettő Commodore 64-re készült, és grafikai segédlet. Esetleg, ha valaki meg akarná szerezni őket, szívesen állok rendelkezésére.

Az utolsó mondat, remélem, azt jelenti, hogy ha valaki boltban megvásárolja a fenti programokat, akkor olvasónk szívesen ad tanácsot a programok megismeréséhez.

Halász Péter, Budapest,

Irinyi u. 42. 1117

Az 1986. márciusi számban a µklub rovaton belül megjelent egy cikk „KOMPUTER HÍREK" címmel. Ebben egyebek között egy, a HCC Commodore szekciójára vonatkozó utalás is van, miszerint az egyetemi klub nem a HCC szekciója. A lapban többször megjelent, hogy a HCC különböző szekciói hol találhatók meg, de az információk tévesnek bizonyultak.

Kérem Önöket, hogy írják meg végre egyértelműen, hogy a különböző klubokat hol lehet megtalálni, és ki a vezetőjük.

A HCC szekciói részére nagyon nehezen tudunk állandó klubhelyiséget találni. A HCC vezetése kéri mindazokat, akik 100—150 fős helyiséget tudnak ajánlani, asztalokkal és néhány zárható szekrénnyel, esetleg térítés nélkül, keressék meg a HCC elnökét (Dr. Simonyi Endre, telefon: 556-245 illetve titkárát (Diebel Dietrich, telefon: 296-600/166), vagy bármelyik szekció vezetőjét. Az egyes szekciók — most — az alábbi helyeken tartják összejöveteleiket:

Aircomp Homelab: Barabás Dezső, telefon: 176-487/22 mellék,
Belvárosi Művelődési Ház, V. Molnár u. 9., minden kedden 6—8 óráig
Apple: Diebel Dietrich, 296-600/166,
Báthori u. 16.; a hónap 1. és 3. keddjén, 3 órakor
Commodore: Dr. Simonyi Endre, 556-245, helyszín változó. Simonyit felhívni.
M 6800: Dr. Simonyi Endre, 556-245, helyszín változó. Simonyit felhívni.
HT: Győző Miklós, telefon: 692-800/név
TRS—80 változó. Győző Miklóst felhívni.
SINCLAIR: Hivessy Ferenc, 403-990, Szellőző Művek, XI., Építész u., hétfő 18—20 óráig

Johonyák Zsolt Csaba, Oradea,

str. Aluminei Nr. 86. BL Sc A op. 18. Románia

A kolozsvári műegyetem gépgyártás-technológia szakos hallgatója vagyok. Nagyon érdekel a számítástechnika és a szoftver. Magam is rendelkezem egy Sinclair ZX Spectrum személyi számítógéppel.

Szeretnék levelezni és programokat cserélni magyarországi Spectrum-tulajdonosokkal.

Remélem, lesznek levelező partnereim.

Juhász József, Alsózsolca,

Ságvári E. út 5. 3571

A Mikroszámítógép Magazin 1986. áprilisi számában Szilágyi Róbert andornaktályai olvasótársamnak a következőket válaszolta:

— A Magazin első száma már aranyért sem kapható!

Nos, én az „aranyat" érő első számot elküldeném cserébe annak, aki küld címemre egy 1984/5-ös számot. Nekem ugyanis csak az hiányzik a sorozatból, az 1983-as bevezető számból kettőt is sikerült utólag keríteni, mert nincs mindenki gyűjtőszenvedéllyel megáldva.

Ezért kérem, az olvasói rovatban levelemet szíveskedjenek közzétenni.

Közzétettük.

Lakatos Péter, Miskolc,

Klapka Gy. u. 36. 3524

A Mikromagazin régi olvasója vagyok. Néhány kivétellel minden szám megvan nekem. Ezek az 1983-as, 84/1, 84/3, 85/5.

Egyre több a színvonalas cikk, de még mindig kevés a hardver és a szoftver. A „Diákrovat" nagyon tetszik, sok újdonság található benne. Írták, a diákszerkesztőség számára tagokat keresnek. Én szívesen lennék „levelező" tagja.

A Mikromagazinnak is lehetne néhány különkiadása, az ÖTLET BIT-LET-jéhez hasonlóan, amiben a válogatott cikkek, programok, hardverötletek és az „Építsünk számítógépet" sorozat jelenne meg!

Nagyon sok tervünk van, pl. ilyen a µ Könyvtár is, ezeknek a megvalósításához nem könnyű az anyagiakat előteremteni. A Diákszerkesztőség vezetője: KARDOS ZSUZSA. Tel: 154-250/574.

Gácsik Attila, Miskolc,

Klapka Gy. u. 56. IV. 3. 3524

1984 januárjától vagyok a µM törzsolvasója, és szívesen olvasom lapjukat. Bár csak hobbiból foglalkozom a számítástechnikával és tudom, hogy ebben a korban (33 évesen) nem annyira fogékony, ill. képlékeny a felfogásom a programozás iránt, mint a most felnövő nemzedéknek, mégis igyekszem minél jobban elsajátítani a programozás fogásait.

Visszatérve az újságra, tetszésemnek adok hangot, ti. a havonkénti megjelenés előnyére válik, például a sorozatok valóban sorozatok lesznek, nem pedig szétszabdalt részcikkek. A lap tetszetős külalakú, a rovatszerkezetét és a cikkek tartalmát illetően is elégedett vagyok. Egyetlen fájó pont (gondolom nem csak nekem), ha túl sok, szinte semmit nem mondó, de annál tetemesebb helyet foglaló reklám található a lapban. Jó, tudom, hogy az hozza a pénzt, de az olvasókat aligha! Egy lap sikerét mégis azon lehet lemérni, hogy hányan olvassák! Talán egy kisebb közvélemény-kutatással el lehetne érni, hogy beállítható legyen a lap hasznos tartalma és a reklámok közötti kompromisszum. És ha már a bírálatnál tartok, hadd kérjek önöktől egy kicsit gondosabb tördelést, ill. „lektorálást", ugyanis elég sokszor találkozom olyannal, hogy az adott cikkhez tartozó gépi kódú lista elkeveredett, vagy egyik (talán a 85/5.) számukban ismét megjelent a HT 1080 Magyar hibaüzenet c. cikk, de a gépi kódú listán egészen más program volt. Ilyesmi könnyen elkerülhető lenne egy hozzáértő által elolvasott „kefe"-levonattal. (Ha ugyan még van olyan.) Csak egy csöpp figyelem kellene, és valóban minőségi újságról beszélnek majd az olvasók, ha szóba kerül a µ M!

Köszönjük a dicséretet és a kritikát is. Amíg nem kerültem egy újsághoz közel, addig én is azt hittem, hogy lehet hibátlan lapot készíteni. Szerkesztőségünk minden egyes alkalommal megpróbálja — sajnos sokszor sikertelenül. Ami a reklámokat illeti, ezek tartják el a lapot, így az arány változása egyenlőre nem várható.

Tóth László, Sopron,

Baross út 8/d 9400

A Mikroszámítógép Magazin című kiadvány alkalomszerű olvasója vagyok.

Rendszeresen olvasom viszont a Markt&Technik Kiadó (NSZK) 64'er és a Happy Computer című újságokat.

Véleményem szerint mindkét, NSZK-ban megjelenő kiadvány tartalmaz olyan írásokat, amelyek a magyar mikroszámítógép-használók érdeklődésére számot tarthatnának.

Ezúton felajánlom önöknek közreműködésemet egyes német nyelvű írások magyarra fordításához.

Küldjön lehetőleg rövid és figyelemfelkeltő írásokat, a külföldi lapban megjelent cikkek kivonatait. Pontosan nevezze meg a forrást, ha új és érdekes írásokat válogat, megjelentetjük.

Sakál Zoltán,

Szolnok 5000

Szerintem — a lap hangvételével ellentétben — egy számítógép nemcsak arra való, hogy programozzam, kiismerjem a memóriafelosztást stb., hanem arra is (és ezt érzem a fontosabbnak), hogy *felhasználjam;* azaz bekapcsoljam, és a bővében lévő felhasználói programok közül a számomra legmegfelelőbbet kiválasztva, a program által értelmesen alkalmazott menü és kommunikációs rendszer segítségével alkalmazzam a programot. (A tv-t sem csak úgy tudom élvezni, ha ismerem az áramköreit, hanem úgy is, hogy bekapcsolom és nézem. Bár kétségtelen, ha „belepiszkálni" — javítani, fejleszteni akarom, akkor ennyi nem elég.) Mindez persze akkor igaz, ha bővében válogathatok a jobbnál jobb programok között.

Szívemből szólt.

Oláh Gyula, Debrecen,

Ispotály u. 15. III. 15. 4025

Örömmel értesültem arról, hogy megjelent az üzletekben a TV-COMPUTER. Felkeltette az érdeklődésemet ez a gép, és szeretnék Önöktől néhány kérdésemre választ kapni.

Eddig SPECTRUM-ot szerettem volna ven-

ni; esetleg C 16-ost. Még csak néhány hónapja ismerkedem a számítástechnikával, s ezért elsősorban olyan gépet szeretnék venni, amelyhez játékprogramok is kaphatók, hisz egy magamfajta számítógép-kedvelőnek fontos az, hogy amíg elfogadható programokat nem tud írni, addig is örömmel használhassa a gépet. (Sokan az első néhány kudarctól elfordulnak a gépektől.) Szóval a lényegre térve: vannak-e, vagy inkább várhatók-e a közeljövőben elfogadható programok a géphez? Kaphatók-e perifériák a géphez, adnak-e ki e típushoz részletesen ismertető könyvet? (A gépkönyvet az üzletben átfutottam, de eléggé felületesnek tűnik.) Elterjed-e önök szerint ez a gép nálunk? Nem kell-e attól tartanunk, hogy az első széria esetleg gyenge minőségű? (Sokaknak okozott ez kellemetlenségeket az AIRCOMP 16-osnál.)

Én beszéltem a Videoton illetékes vezetőivel, akik azt mondták, hogy viszonylag rövid idő alatt a TV-Computer-t minden szempontból (ár, programellátás, vevőszolgálat stb.) versenyképessé teszik a nálunk kedvelt külföldi gépekkel. Levelét elküldtem a VT kereskedelmi igazgatójának.

Honti József, Csákvár,

Május 1. u. 11. 8083

Mint már egy korábbi levelemben beszámoltam róla, községünkben is megalakult a Mikroklub. A klub még nincs egyéves, de már bizonyítja életképességét. Különösen a fiatalok körében nagy az érdeklődés a klubmunka iránt.

Örülök a klub megalakulásának, a küldött programot átadtam az illetékes rovatvezetőnek.

Köszönöm a leveleket, valamennyi olvasónknak kellemes nyaralást kíván

KOVÁCS GYŐZŐ

Olvasónk javasolja

A legnépszerűbb mikrogépeknek (jelenleg a Commodore, a Spectrum, a HT és a Primo) nyisanak egy-egy dupla oldalt, ami programoktól kezdve hírekig mindent tartalmazhatna, de csak azzal a számítógéppel kapcsolatban. Szerintem ugyanis lapjuk vásárlóinak zömét éppen ezek programozói alkotják.

Ennek a két oldalnak a kezelését rá lehetne bízni egy-egy működő klubra. Ők felhívásokat intézhetnének az adott gép felhasználóihoz, hogy küldjenek be saját készítésű, általuk érdekesnek tartott programokat. Ezeket ők maguk tesztelhetnék, a szerzője engedélyével módosíthatnák, esetleg bővíthetnék. Sokakat a cikk megírása riaszt el a publikálástól. Helyettük vállalhatnák ezt a munkát is. Ez a megoldás tehermentesítené a lap dolgozóit, és lehetővé tenné több program megjelenését, ami — ha szabad környezetemből kiindulva általánosítanom — az olvasótábor kívánsága.

PIVARNYIK ATTILA

Az ötletet jónak tartjuk, várjuk a vállalkozó klubok jelentkezését. A szerkesztőség.

Árak és vámok

Legutóbbi levelemből a 86/1. számban a reméltnél többet, de kevesebbet is közöltek.

Ugyanis az ÁRAK miatti háborgásom Kovács Győző kedves, csitító szavai ellenére nem szűnt meg, de azt hiszem, öbenne sem.

Idézet a 86/2. szám µ'86 kiállítás bevezetőjéből: „. . . a számítástechnika tömeges elterjedésének még ma is az egyik legnagyobb akadálya a magas ár."

Hozzátenném, a *bűvészkulcsokkal* kalkulált hazai ár. Példák:

— Az Ötletben Angyalosi László még ma is várja tán, hogy jelentkezzenek azok a cégek, amelyek neki — és az olvasóknak — meg tudják magyarázni, miként lehet az ÁPISZ-nál 8000 Ft alatt haszonnal értékesíteni a C—16-os gépeket, másoknál viszont csak 10-12 ezer Ft-ért.

— Az SFD 1001 (1,05 Mb-os) lemezmeghajtót a Centrum Nagyker. V. a kisker. egységeknek egy C—610-es alapgépre épülő konfiguráció tagjaként *49 500* Ft-ért árulja, ugyanakkor a Vámhivatal az ajándék- és utasvámtarifa alapján *60 000* Ft vámalappal számol, és — most jön a bűvészkulcs — a Novotrade 2C áruház telefoninformációja szerint ők *150 000* Ft-ért árusítják! A nyugati árát nem ismerem, csak sejtem, hogy a C—128 mellett árult hasonló gép 7-8000 osztrák schillingjéhez nem állhat messze. (A Herlango téli katalógusában 7990,— Sch)

Kovács Győző azzal biztatott, hogy ezek realitások! Ahogy vesszük.

Egy közületnek nem jelent akkora különbséget, hogy valami, amit meg akar valósítani az 100 000 vagy 150 000 Ft-jába kerül, így mint piaci realitást könnyen elfogadja az így képzett árakat. Ennyiben tehát realitás.

Az ilyen piacon lezajló események azonban igen messze állnak a számítástechnika valódi tömeges elterjesztésétől. Legfeljebb akkor, ha már mint leírt eszközök kerülnek majd mondjuk ajándékozás útján a klubokhoz (5-10 év múlva).

A sajtó szavával viszont igen eredményesen lehet — s szerintem kell is — küzdeni a „bűvészkulcsos" árképzés ellen, például az ilyen árképzés nyilvánosságra hozatalával! Tán várjuk meg a sajtótörvényt?

A sajtónak az ilyen magatartása már eredményesen megmutatkozott — a kormányszerveket is döntésre késztetve —, amikor pl. a vámkulcsokat a mikrók és videók körében jelentősen, de sokak szerint még nem elégségesen lecsökkentették.

A Vámőrség parancsnoka a HVG-ben nyilatkozott úgy, hogy nem a vámok verik fel az árakat. Hát ez igaz is, meg nem is. A kialakuló — bizományi — árakban még ma is jelentős tételnek szerepel a vám. A vámhivataloknál belső használatra kiadott lista, mely a belföldi forgalmi érték — vámalap — meghatározásait a legismertebb árukra tartalmazza, egyrészt nem vagy igen ritkán kerülnek nyilvánosságra, ugyanakkor ritkán kerülnek változtatásra. Ezeket a változtatásokat pedig a nyugati kiskereskedelmi árakban csaknem hetente lehet tapasztalni, de egy-egy konkrét géptípus esetén félévenként valóságos „felezési idő" valósul meg. A Sinclair Spectrum például közel egy árban volt a C—64-gyel még egy évvel ezelőtt — cca. 4000 schilling —, most 1500-1700 Sch. A vámalap viszont még tudomásom szerint most kb. 21 000 Ft. (kiskereskedelemben 14-16 000 Ft-ért is látni).

Az összefüggés a belföldi árakkal pedig nyilvánvaló. Ha a vám 21 000 Ft után kerül kiszabásra, az utasforgalomban behozott gépet — állami szervezett import alig van — a tulajdonos nem fogja, ill. nem akarja ennél olcsóbban eladni.

Tehát ha a vámszervek által használt lista nem követi folyamatosan a külföldi árcsökkenéseket, nem fogja engedni ennek jótékony begyűrűzését sem! Így már nem igaz a Vámőrség Országos Parancsnokság állítása sem, a vámok áralakító hatásáról.

Nem bonyolódom bele a valutaárfolyamon történő számítások kérdésébe, mert az igaz, hogy esetenként az exportban nem lehet az árfolyamokat érvényesíteni, viszont a valutainség miatt az így létrejövő üzletkötésekre is szükség lehet.

A vámot, illetve a devizagazdálkodást érinti az a kérdés is, hogy az utas 25 000 Ft belföldi forgalmi értékhatárig hozhat árut. Ez sem követi egyébként a VI. ötéves tervben megvalósult 4,3—8,8%-os inflációt, de a fentebb írt lassú „listaváltozás" folytán előállt a múlt évben egy olyan helyzet, hogy a jogilag nem kifogásolható, de a közvélemény megítélése szerint annál inkább kifogásolt módon a C—64-es gépet behozók — a vámalap akkor 27 000 Ft volt — nem kapták meg az utólagos behozatali engedélyt, s gépük kényszereladásra került, mégcsak elővételi jogot sem biztosítva részükre. (A Heti Világgazdaság olvasói rovata is írt ilyenekről.) Ezt kivédendő születtek azok az „ajándékozó" okiratok, melyek leegyszerűsített eljárásban és magasabb értékhatárral vitték a jogon kívüli útra az embereket. Embereken nem a spekulánsokat, hanem azokat értem, akik saját fejlődésük, rajongásuk érdekében döntöttek az utazásuk alatt a gépvásárlás mellett.

A spekulációk visszafogása véleményem szerint sokkal inkább ez utóbbi módszernél kellene, semmint a 25 000 Ft-os értékhatár változatlanul hagyásával. A fent már írt listamerevség folytán könnyen elképzelhető, hogy a szabályosan szerzett valuta felhasználásával — illetve annak kb. 50%-a ami vásárlásra fordítható — ennél magasabb belföldi forgalmi értékű árut tudunk beszerezni.

LUKÁCS ATTILA

Keresünk megvételre jó állapotban lévő

VIDEOTON gyártmányú
VDT 52103, VPC, VPPC
display-ket.

Villamosenergiaipari
Kutató Intézet
Ügyintéző: Papp György osztályvezető
Telefon: 178-698

Könyvek

Newman, W. M.— Sproull, R. F.: Interaktív számítógépes grafika (Budapest, 1985. Műszaki Könyvkiadó, 492 oldal. Ára: 165,— Ft.)

A számítógépes grafika kapcsolatot teremt a számítógép és az ember között: a számítógép egy grafikai képet állít elő sornyomtatóval papíron vagy a megjelenítő képernyőjén az ember számára, amelyet az vizuálisan észlel. A nemzetközileg is elismert szerzőpáros kiváló szakkönyve részletesen tárgyalja a számítógépes grafika hardver- és szoftvereszközeit, a használatos számítástechnikai és matematikai módszereket, és foglalkozik az ilyen rendszerek fejlesztésével és alkalmazási lehetőségeivel.

Vitray Péter: Hetedhét Atari 800LX (Budapest, 1985. Novotrade RT. 151 oldal. Ára: 92,— Ft.)

A Hetedhét . . . című sorozat célja, hogy a különböző számítógéptípusokat a teljesen kezdő felhasználók számára érthető nyelven, szórakoztató formában, szellemes, egyszerű programok segítségével mutassa be — mindössze hét nap alatt. E célját a könyv sikerrel teljesíti, amikor a gép üzembe helyezésétől kezdve, a programírás alapismeretein keresztül, a BASIC nyelvű utasítások gyakorlásával a felhasználókat egyszerű játékprogramok írására is képessé teszi. A kötetet a gép használatát és a programírást segítő táblázatok egészítik ki.

Pomper János: Példák a mikroszámítógépek vállalati alkalmazására (Budapest, 1985. COMPORGAN, 375 oldal. Ára: 370,— Ft.)

A kötet a vállalatok vezetési, szervezési, irányítási feladatainak mikroszámítógépes megoldásaival foglalkozik. 27 oktatási célú programot tartalmaz, melyek egy 60 modulból álló, bármely mini- vagy mikroszámítógépre implementálható, ipari alkalmazásra készült programcsomag erősen leegyszerűsített változatai.

A szerző a világ 500 legnagyobb vállalatánál végzett nemzetközi felmérés eredményét hasznosította annak megállapítására, hogy milyen vezetési problémák megoldásában használnak leggyakrabban számítógépet, és milyen módszereket alkalmaznak a problémák megoldásában. Az egyes fejezetek a megoldások módszereit tárgyalják.

Végül a szerző a modellalkotás legfontosabb kérdéseit foglalja össze, és táblázatot közöl a programok méreteiről is.

Személyiszámítógép-javítás, karbantartás közületeknek, magánszemélyeknek.
Egyedi megrendelés alapján kiegészítő berendezések gyártása.
Pl. Sinclair fényceruza, Joystick Interface, oktatási intézménynek kabinet kialakítása.

Pásztor Ferenc személyiszámítógép-javító és -karbantartó kisiparos. Szolnok, Mátyás király u. 2. V/3.

Nemzetközi Számítástechnika — Internacia Komputado

Az 50 országban olvasott magazin példányonként 30 forintért kapható az SKV Könyvesboltban

Bp. II., Keleti Károly u. 10. Telefon: 158-018

A NOVOTRADE RT

az alábbi szoftverek bemutatóját tartja a Hotel Rege

„KÁRTYA" termében
(Bp. II. ker., Pálos u. 2.)
de. 10 órától,
melyre
minden érdeklődőt
szeretettel várunk!

IX. 1.

Főkönyv- és folyószámla-könyvelési rendszer bemutatása

IX. 2.

Készletgazdálkodási rendszer

IX. 3.

Bér- és munkaügyi rendszer
Személyzeti rendszer

IX. 4.

Alap-, illetve fejlesztői rendszerek
és lokális hálózatok
IBM kompatíbilis
gépekből kialakítva

IX. 5.

Állóeszköz-gazdálkodás

NOVOTRADE

Kínai szénbányákban

Négyezer darab professzionális mikroszámítógépet vásárol Kína az amerikai Intel cégtől. Ezeknek az IBM PC-vel kompatibilis gépeknek csupán az egynegyede érkezik összeszerelt állapotban. A háromezer komplett alkatrészkészletet a Hongkong mellett fekvő sencsieni különleges gazdasági övezet egyik gyárában szerelik össze. A gépeket elsősorban a bányászati ágazatban, ezen belül is a szénbányászatban kívánják alkalmazni.

10 Mbájtos tároló áramkör?

A Japán Tudományos és Technológiai Ügynökség háromféves kutatási tervet indított, hogy kifejlesszenek egy olyan nagy sebességű memóriamorzsát, amely helyettesítheti a hagyományos mágnesszalagot és mágneslemezt. A 10 Mbájt tárolási kapacitású SRAM integrált áramkört a Citizen óragyár fejleszti a japán kormány 2,5 millió dolláros hozzájárulásával. Az első kereskedelmi példányok megjelenése 1990-re várható, és körülbelül 100 dollárba fognak kerülni.

A szakemberek szerint az új tároló forradalmasíthatja a hordozható mikroszámítógépek piacát, ahol a lemezorientált gépek nem tudnak igazán meggyökeresedni.

Szimat-robot

A fejlett ipari robotok már „látnak" és „hallanak". Az oxfordi Austin autógyárban viszont már olyat is kifejlesztettek, amely „szaglászik", s jelzi is az észlelt szag helyét. A robot feladata annak megállapítása, hogy jól zárnak-e az új kocsik. Először héliumgázt fecskendeznek a kocsikba, és ahogy azok a futószalagon továgördülnek, a roboton lévő gázérzékelő „végigszagolja" a karosszériát, elsősorban az ajtók és az ablakok mentén.

Iskolagépek a Szovjetunióban

A reál tagozatos szovjet tízosztályos iskolákban már 5-10 db iskolaszámítógép működik. Ezek részben szovjet gyártmányú Agatok, részben japán Yamaha háziszámítógépek.

Mint ismeretes, az Agat gépet 1983 novemberében jelentették be a Szovjetunióban; ez az Apple II funkcionális megfelelője. 1985 végére már megtörtént a gyártásba vétele, de sajnós mágneses háttértár nélkül. Így ezeken a gépeken egyelőre csak „pötyögni" lehet.

Az Elektronorgtechnika tavaly hirdetett nemzetközi versenytárgyalást az iskolaszámítógép importjára, amelyen nagy esélyesként részt vett a Sinclair ZX-Spectrum gép is. A győztes — óriási meglepetésre — a Yamaha nevű japán háziszámítógép lett. Az értesülések szerint a 8 bites gépet képernyővel, hajlékonylemezes tárolóval és nyomtatóval 314 USA-dollárért szállították a japánok. Az üzlet érdekességét növelte, hogy az amerikai Microsoft cég által kidolgozott, a különböző típusú számítógépek közötti teljes kompatibilitást megteremtő MSX (Microsoft Extended Basic is) operációs rendszer fut a gépeken. A jelentések szerint az idén összesen 4000 db Yamaha-típusú gépet szállít a Nippon Gakki cég a Szovjetunióba.

A Proper 8 és 16 után 32

Már a 16 bites IBM PC-vel kompatibilis gépeket „kinövő" felhasználókat készíti fel a 32 bites gépre való átállásra az SZKI új „kétarcú", Proper 16/32/MT-típusú személyi számítógépe. A 16 bites mikroprocesszor biztosítja a gépnek a Proper 16-tal való kompatibilitását, hiszen fut rajta a Propos operációs rendszer 3.2 verziója, a 32 bites mikroprocesszorán pedig a Unix operációs rendszert használják. Ez utóbbinak 16 Mbájtos címezhető memóriaterülete van, a valós tár is 0,5-2 Mbájt között változik. A gép műveleti sebessége mintegy háromszorosa a Proper 16-énak.

Az új géptípus jól hasznosítható a többfelhasználós ipari és mezőgazdasági alkalmazásoknál, a mérnöki tervezői munkahelyeken és az irodaautomatizálási feladatoknál.

Morze adó-vevő

ZX-Spectrum gépre dolgozta ki a HG5VQ hívójelű Csapó István a morze adó-vevő programját. Az *adás* sebessége tetszőlegesen beállítható 30-150 karakter/perc intervallumban. A program a begépelt és a képernyőn lévő karaktereket morzejelekké alakítja. Az adni kívánt szöveget folyamatosan gépelhetjük. *Vétel* esetén a program folyamatosan méri az adóállomás adási ütemét, és ennek megfelelően módosítja a vételhez szükséges időadatokat. Viszonylag hamar rááll a megfelelő sebességre, és ott megbízhatóan dolgozik.

Telexkapcsolat a számítógéppel

Sok vállalat a telexgépet nemcsak üzenetváltásra használja, hanem feldolgozandó adatot is küld át. Az eddigi gyakorlat az, hogy a fogadó cégnél valaki számológéppel dolgozza fel ezeket az adatokat.

A Háziszámítógép-építők Klubja erre a feladatra elkészítette a ZX-Spectrum és a Commodore 64 gépre a megfelelő programot. Első része az úgynevezett „lefülelő szoftver", ami a telexgépen érkező adatokat automatikusan, párhuzamosan a számítógépbe küldi. Ezen az üzenetváltás befejezésekor az eredmény rögtön megjelenik. Emellett lehetőség van arra is, hogy a telexgépről lejövő lyukszalagot dolgozza fel a gép.

MEGJEGYEZZÜK...

A lap ez évi 7. számában „Jelentkezem" címen Baky Miklós tollából egy rövid humoros írást közöltünk. Az olvasók leveleiből úgy tűnik, hogy az írást sokan félreértették, és úgy hitték, hogy ezzel a dr. Simonyi Endre által kiírt pályázatot akarjuk lejáratni.

A szerkesztőség kijelenti, hogy ez félreértés. A lap az amatőrök szellemi termékeinek értékesítését elősegíteni kívánó pályázatot nem akarta támadni, sőt sajnálatosnak tartja, hogy eddig mindössze 8 pályázat érkezett, amelyből kettő látszik olyannak, amiből „talán valami lesz" (dr. Simonyi közlése). Úgy véljük, hogy az ilyen pályázatokhoz még nem szoktunk hozzá, de hát minden kezdet nehéz.

A szerkesztőség

COMMODORE-tulajdonosok figyelmébe!

— **C16, C116 RAM memóriájának bővítése 64 k-ra beépített bővítővel: BASIC munkaterület 60671 bájtra nő**
— **videoadapter beépítése C16-ba**
— **magyar betűkészlet beépítése C16, C116 és PLUS/4-be.**

Iskoláknak engedmény!
Vidékieknek egy utazással!
Kérje részletes tájékoztatónkat!
EL—KA GM. 346-933 délután.
Levélcím: 1141 Budapest, Lipótvár u. 59.

EL—KA Elektromos Kapuszerviz és Automatika
Tervező, Kivitelező Gazdasági Munkaközösség
1141 Budapest, Lipótvár u. 59. Németh Tibor, 574-811/192.

PRODUKTORG
SZERVEZÉSI VÁLLALAT

C64, M08X, IBM PC XT kompatibilis rendszerekre ajánljuk programcsomagjainkat:

— főkönyvi könyvelés,
— költségfelosztás,
— állóeszköz-nyilvántartás,
— rendelés-nyilvántartás,
— bérelszámolás,
— adóelszámolás.

1027 Budapest, Fő u. 68.
Telefon: 154-090

Bővebb felvilágosítással készséggel állunk rendelkezésükre!

SZÁMÍTÓKÖZPONT-VEZETŐK!

Ne dobják ki sérült mágneslemezeiket!
Gyorsan,
olcsón,
6 havi garanciával megjavítjuk!
Commodore 64 számítógépükhöz
— az adatrögzítést megkönnyítő —
kiegészítő numerikus klaviatúrát
raktárról szállítunk,
1 éves garanciával.
Megvásárolható
az Econorg
1. sz. számítástechnikai
szaküzletében:
Bp. VI., Szinyei Merse Pál u. 1.
Tel.: 127-628

Egészségügyi Elektronikai Gm
1045 Bp., Erzsébet u. 14. X. 59.
Telefonügyelet: Grósz Andor, 632-720 (9—17 óráig)

ORTEKON GMK

ORTEKON TERVEZÉSI ÉS FEJLESZTÉSI gazdasági munkaközösség
1036 Bp., Árpád fejedelem útja 69.
Telefon: 154-250, 887-861
Bányai

Vállaljuk:

— műszerek, villamos és elektronikus ipari termékek konstrukciós és technológiai tervezését, komplex felszerszámozását a sikeres „0" sorozatig;

— elektronikus (digitális) vezérlőegységek, szabályzókészülékek és -rendszerek tervezését és gyártását;

— a gyártás-előkészítés és elszámolás irányítási és adatfeldolgozási feladatainak megvalósítását több munkahelyes számítógéprendszer alkalmazásával.

— CNC-vezérlés komplett felújítását;

A megfelelő referenciákról személyesen adunk tájékoztatást.

DÖNTÉSÉT MEGKÖNNYÍTJÜK!

Széles választék, kedvező ár, sokoldalú szolgáltatás!

8 bites ALAPKONFIGURÁCIÓK

PROPOS—8 (CP/M vagy ezzel kompatíbilis) operációs rendszerrel

M08X	**250 000,— Ft**
PROPER—8	**250 000,— Ft**

A konfigurációk érdekeltségi alap nélkül megvásárolhatók

16 bites ALAPKONFIGURÁCIÓK

PROPOS—16 (IBM PC XT vagy ezzel kompatíbilis) operációs rendszerrel

PROPER—16/m	**266 000,— Ft**
PROPER—16/W1	**439 000,— Ft**
PROPER—16/W2	**549 000,— Ft**
PROPER—16/MEGA	**699 000,— Ft**
MÁTRIXNYOMTATÓ (MP—80, kvázigrafikus jelkészlettel)	**30 000,— Ft**

RENDELÉSEKET VESZÜNK FEL 1986. II. FÉLÉVRE

PROPER—16/MT (IBM PC AT kompatíbilis operációs rendszer)
850 000,— Ft

PROPER—16/G (nagy teljesítményű grafikus munkahely)
1 500 000,— Ft

SZOLGÁLTATÁSOK:

12 hónap garancia, további 12 hónapra a hardver árának 6%-áért
hardver-szervizszolgálat
Világszínvonalú perifériák
Széles körű szoftverválaszték
Kulcsrakész alkalmazói rendszerek
Hálózatba integrálási lehetőség
Országos szervizhálózat

KÉRJE RÉSZLETES ÁRJEGYZÉKÜNKET!

Szki

Számítástechnikai Kutató Intézet és Innovációs Központ
Budapest 1251 Pf. 19.

Információ:
SCI—L Számítástechnikai Informatikai Fejlesztő Leányvállalat
Budapest, Iskola u. 10. 1011
Telefon: 153-204

AZ SZKI STABIL PARTNER!

SCITEL Számítástechnikai Fejlesztő Leasing Leányvállalat
Budapest, Donáti u. 35—45. 1015
Telefon: 350-180/142

Várjuk Önöket az ŐSZI BNV F 2-es pavilonjában!
Keresse a FOTOELEKTRONIK EMBLÉMÁT!

FOTO-ELEKTRONIK

```
10 REM  * * * * * * * * * * * * * EZ C64 PROGRAM * * * * * * * * * * * * * *
20 REM                            * * * * * * * * * *
30 B$="                           SYSTEM SZERVEZÉSI VÁLLALAT
40 C$="                           1055 BUDAPEST, SZENT ISTVÁN KRT. 11.
50 REM  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
60 PRINT" ♥ ":PRINT TAB/1/:"      MUNKAHELYE DOLGOZHATNA SZERVEZETTEBBEN?
70 REM  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
80 PRINT: TAB/5/:"                I (IGEN), N (NEM) A BEÍRANDÓ!
90 REM  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
100 GET A$:IF A$="N" THEN 140
110 IF A$="I" THEN 180
120 GOTO 100
130 REM * * * * * * * * * * * * * HA N (NEM) * * * * * * * * * * * * * * * * *
140 PRINT" ♥ ":PRINT TAB/5/:"     REMEKI EZ A LEGJOBB MUNKAHELY!
150 PRINT TAB/4/:"                HÍVJANAK MEG TAPASZTALATCSERÉRE!
160 PRINT B$:C$:FOR I=1 TO 1E3:NEXT I:GOTO 140
170 REM * * * * * * * * * * * * * HA I (IGEN) * * * * * * * * * * * * * * * * *
180 PRINT" ♥ ":PRINT TAB/14/:"    AKKOR AJÁNLJUK
190 PRINT TAB/5/:"                MIKROSZÁMÍTÓGÉPPEL TÁMOGATOTT
200 PRINT TAB/7/:"                MUNKA-, ÜZEM-, VÁLLALAT- ÉS
210 PRINT TAB/6/:"                INFORMÁCIÓ RENDSZERSZERVEZÉSI
220 PRINT TAB/3/:"                -VEZETÉSI (DÖNTÉSELŐKÉSZÍTÉS STB.)
230 PRINT TAB/4/:"                -GAZDÁLKODÁSI (ANYAG-, MUNKAÜGY-,
240 PRINT TAB/0/:"                ÁLLÓESZKÖZ-, MENETLEVÉL-FELDOLGOZÁSI STB.)
250 PRINT TAB/15/:"               -TERMELÉSI
260 PRINT TAB/4/:"                (BERUHÁZÁSI, KARBANTARTÁSI STB.)
270 PRINT TAB/1/:"                MÓDSZEREINKET, RENDSZEREINKET, PROGRAM-
280 PRINT TAB/6/:"                CSOMAGJAINKAT ÉS MIKROGÉPES
290 PRINT TAB/5/:"                PROGRAMOZÁSI SEGÉDESZKÖZEINKET.
300 PRINT TAB/2/:"                OSZTOTT ADATFELDOLGOZÁSI RENDSZEREK
310 PRINT TAB/6/:"                TERVEZÉSÉT, FEJLESZTÉSÉT ÉS
320 PRINT TAB/6/:"                ÜZEMELTETÉSÉT IS VÁLLALJUK.
330 PRINT TAB/6/:"                REFERENCIÁKKAL RENDELKEZÜNK!
340 PRINT TAB/0/:"                RÉSZLETESEBB ISMERTETÉS E LAP OLDALÁN
350 PRINT TAB/2/:"                TALÁLHATÓ. AZ ISMERTETETT MÓDSZEREK,
360 PRINT TAB/5/:"                RENDSZEREK, PROGRAMCSOMAGOK ÉS
370 PRINT TAB/7/:"                ESZKÖZÖK MEGTEKINTHETŐK AZ
380 PRINT TAB/4/:"                ORGTECHNIK HUNGÁRIA BUDAPEST '86
390 PRINT TAB/3/:"                SZAKKIÁLLÍTÁSON, AHOL SZAKEMBEREINK
400 PRINT TAB/2/:"                AZ ÉRDEKLŐDŐK RENDELKEZÉSÉRE ÁLLNAK.
410 REM * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
420 FOR I=1 TO 5E4:NEXT I:PRINT"  ":PRINT B$:C$:FOR I=1 TO 1E4:NEXT I:GOTO 140
430 REM                           LEVÉLCÍM: 1363 BP. 502. PF. 33
440 REM                           TELEFON: 126-670
```